

# CUBIT Mesh Generation Environment

## Volume 2: Developers Manual

Cubit Development Team<sup>1</sup>  
Sandia National Laboratories  
Albuquerque, New Mexico 87185

### Abstract

The CUBIT mesh generation environment is a two- and three-dimensional finite element mesh generation tool which is being developed to pursue the goal of robust and unattended mesh generation—effectively automating the generation of quadrilateral and hexahedral elements. It is a solid-modeler based pre-processor that meshes volume and surface solid models for finite element analysis. A combination of techniques including paving, mapping, sweeping, and various other algorithms being developed are available for discretizing the geometry into a finite element mesh. CUBIT also features boundary layer meshing specifically designed for fluid flow problems. Boundary conditions can be applied to the mesh through the geometry and appropriate files for analysis generated. CUBIT is specifically designed to reduce the time required to create all-quadrilateral and all-hexahedral meshes. This manual is designed to serve as a reference and guide to creating finite element models in the CUBIT environment.

---

1. See the next page for the members of the CUBIT Development Team.

## Cubit Development Team Membership

### Sandia National Laboratories, Albuquerque New Mexico

Brett W. Clark

Randy R. Lober      Advanced Engineering & Manufacturing Software

Scott A. Mitchell      Applied & Numerical Mathematics

Ray W. Ostensen

Gregory D. Sjaardema      Engineering and Manufacturing Mechanics

Timothy J. Tautges      Computational Mechanics & Visualization

David R. White

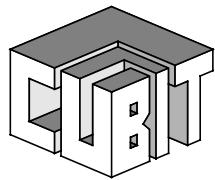
Tammy J. Wilson      Computational Mechanics & Visualization

### Brigham Young University, Salt Lake City, Utah

Steve Benzley      Civil Engineering Department

### Consultants

Malcolm Panthaki



---

# Table of Contents

<b>Table of Contents</b>	<b>iii</b>
<b>Class Hierarchy</b>	<b>xi</b>
<b>Introduction</b>	<b>1</b>
State of the Manual .....	1
About the CUBIT Project .....	2
About this Manual .....	4
Problem Reports and Enhancement Requests .....	6
Additional Information .....	7
<b>Recommended Practices for CUBIT Code Development</b>	<b>11</b>
Style Conventions .....	11
Filenames Conventions .....	12
Documentation Comment Conventions .....	13
Options for Producing Class Documentation .....	15
Summary .....	16
References .....	16
<b>CUBIT Code Features</b>	<b>17</b>
Internal Message Reporting .....	17
AutoTester .....	19
<b>Overview of CubitObject Classes</b>	<b>23</b>
Derived Classes .....	23
The Model Class .....	24
<b>Overview of GeometryEntity Classes</b>	<b>25</b>
Geometry Definition .....	25
<b>Overview of MeshEntity Classes</b>	<b>29</b>
Definitions .....	29
EdgeUses and FaceUses .....	30
NodeHex Description .....	31
<b>Overview of Tool Classes</b>	<b>33</b>
Incomplete at this time .....	33
<b>MeshTool Overview</b>	<b>35</b>
Summary .....	35

<b>DLLList Implementation</b>	<b>39</b>
Implementation Details .....	39
Efficiency Details .....	40
Creating a Variant of a DLLList .....	44
DLLList variants used in CUBIT .....	45
<b>Overview of the User Interface and Graphics Related Classes</b>	<b>47</b>
The Parser Class .....	47
Abstract Command Classes .....	48
Adding Code to Support a New Command .....	48
Future Command Interface Development .....	49
<b>Miscellaneous Classes and Constants</b>	<b>51</b>
CubitVector .....	51
CubitPlane .....	52
AcisGeometryEngine Description .....	55
ArrayBasedContainer Description .....	63
ATTRIB_CUBIT_OWNER Description.....	67
attrib_gtc.hpp Description.....	69
ATTRIB_GTC_NAME Description.....	71
ATTRIB_PARENTS Description.....	73
attrib_snl.hpp Description.....	75
AttributeCommands Description .....	77
AutoVertexType Description.....	79
BaseHexer Description .....	81
BladeEdges Description.....	89
BladeEdgesTree.hpp Description .....	91
BoundaryLayer Description.....	93
BoundaryLayerEdge Description.....	95
BoundaryLayerFace Description .....	99
BoundaryLayerTool Description .....	101
CleanUp Description.....	105
CommandHandler Description .....	115
ControlPoint Description .....	119
CopiedData Description.....	121
CpuTimer Description .....	123
CubitBox Description .....	125
CubitCollection Description .....	127
CubitContainer Description .....	129
CubitDefines Description.....	131
CubitEdge Description.....	133
CubitEntity Description .....	137
CubitFace Description .....	139
CubitHex Description .....	143

CubitKnife Description .....	147
CubitLogical Description .....	151
CubitMatrix Description .....	153
CubitMessage Description .....	155
CubitNode Description .....	161
CubitNodeArray1D Description .....	167
CubitNodeArray2D Description .....	169
CubitNodeArray Description .....	170
CubitPlane Description .....	171
CubitPtrArray Description .....	175
CubitPtrArray2D Description .....	176
CubitPtrArray3D Description .....	177
CubitSheet Description .....	179
CubitStack Description .....	181
CubitString Description .....	183
CubitVector Description .....	185
CubitVoxel Description .....	191
CurveSubDomain Description .....	195
Database Storage Size Constants .....	197
DLLList Description .....	199
DoubletPillower Description .....	205
DoubletPillowerTD Description .....	211
DrawingTool Description .....	215
DrawingToolInstance Description .....	227
DynamicArray Description .....	245
DynLoopList Description .....	247
EdgeMeshTool Description .....	249
EdgeUse Description .....	251
ElementBlock Description .....	253
ElementBlock1D Description .....	257
ElementBlock2D Description .....	259
ElementBlock3D Description .....	261
ElementQuality Description .....	263
ExodusMesh Description .....	265
Face Type (Plain or Whisker) Constants .....	271
FaceUse Description .....	273
FastqEntity Description .....	277
FastqCommands Description .....	283
FileCommands Description .....	285
FREDTool Description .....	287
FullHex Description .....	289
function_templates.hpp Description .....	293
GenesisCommands Description .....	295
GenesisEntity Description .....	297
GenesisTool Description .....	299
GeometryCommands Description .....	303

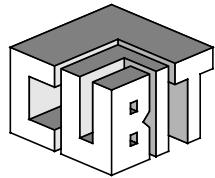
GeometrySizingTool Description .....	307
GeometryTool Description .....	309
GetLongOpt Description.....	319
GraphicsCommands Description .....	321
GridSearch Description.....	323
Hexer Description .....	329
HexKnowingNode Description.....	339
HexQuality Description .....	341
HexQualityController Description.....	343
HexTool Description.....	345
HexToVoid Description.....	347
IntervalLinearProgram Description .....	351
LinearProgram Description.....	357
Loop Description .....	363
LoopCollapser Description .....	365
LoopInterval Description .....	369
LoopJoiner Description.....	371
MappingFace Description.....	373
MapToolSupport Description .....	377
MemoryAllocation.hpp Description .....	379
MemoryBlock Description.....	381
MemoryManager Description .....	383
MeshEntity Description .....	387
MesherCommands Description.....	389
MeshIntersect Description .....	391
MeshTool Description .....	395
Model Description .....	397
ModelCommands Description .....	403
ModelViewer Description.....	405
Mouse.hpp Description .....	407
MuseOutput Description.....	409
NodeHex Description.....	411
NodeSet Description .....	415
ParallelMeshTool Description .....	419
parents.hpp Description .....	421
Paver Description.....	423
PillowNode Description.....	431
PillowSheet Description.....	433
ProtoHex Description.....	437
Pyramid Description .....	443
PyramidTool Description.....	445
QuadQuality Description .....	447
QuadQualityController Description.....	449
QualityCommands Description.....	451
QualityController Description .....	453
QualitySummary Description .....	455

Queue Description .....	457
RefBody Description .....	461
RefEdge Description .....	465
RefEntity Description .....	471
RefEntityName Description .....	479
RefEntityNameMap Description .....	481
RefFace Description .....	483
RefGroup Description .....	495
RefVertex Description .....	497
RefVolume Description .....	501
RotateTool Description .....	509
SDLLList Description .....	511
SelfCrossing Description .....	515
SelfCrossingLoop Description .....	517
SheetChord Description .....	519
SheetChordPoint Description .....	525
SheetEdge Description .....	535
SheetEdgeTree.hpp Description .....	537
SheetEdgeTreeQueue.hpp Description .....	539
SideSet Description .....	541
SizingTool Description .....	545
SLLList Description .....	547
SLLList Description .....	548
SLLListIterator Description .....	549
StairTool Description .....	551
StarNode Description .....	555
StcEdge Description .....	557
Strider Description .....	561
SubDomain Description .....	563
SubMapNode Description .....	565
SubMapTool Description .....	569
SurfDistributeTool Description .....	573
SurfMapTool Description .....	575
SurfMeshTool Description .....	579
SurfMorphTool Description .....	585
SurfSubDomain Description .....	587
SurfVertexTree.hpp Description .....	591
SurfVertexType Description .....	593
SweepTool Description .....	595
TDCellIndex Description .....	597
TDCenterNode Description .....	599
TDCopied Description .....	601
TDDistance Description .....	603
TDGenesisID Description .....	605
TDGeometrySizing Description .....	607
TDHexKnowing Description .....	609

TDLaplace Description .....	611
TDLayer Description .....	613
TDLPEdge Description.....	615
TDMorph Description.....	619
TDNodeHardLine Description.....	621
TDNormal Description .....	623
TDPaver Description .....	625
TDPillow Description .....	627
TDProjection Description .....	631
TDSizing Description .....	633
TDStrider Description.....	635
TDSubMap Description .....	637
TDTipton Description.....	643
TDUVSpace Description .....	647
TDVolMapTool Description.....	649
TDWeight Description.....	651
TimeVal Description.....	653
AbsTime Description .....	654
DeltaTime Description.....	655
Tool Description .....	657
ToolData Description.....	659
ToolDataUser Description .....	663
TranslateTool Description .....	665
Tree Description.....	667
TriangleTool Description.....	671
TriElementTool Description .....	673
TwoHalfDimTool Description.....	675
UserInterface Description .....	681
Database Validity Check Functions.....	685
VolMapTool Description .....	687
VolMeshTool Description .....	691
VolSubMapTool Description.....	693
VolumeEdgeType Description.....	699
VolVoidBoundary Description .....	701
VolVoidSepTool Description .....	703
VoxelTool Description.....	705
WhiskerCell Description.....	709
WhiskerChord Description .....	713
WhiskerFace Description.....	723
WhiskerHex Description.....	727
WhiskerKnife Description .....	735
WhiskerSheet Description .....	739
WhiskerWeaver Description .....	753
XpatchTool Description.....	765
<b>Class Owner/Author Index</b>	<b>767</b>
<b>Class Index</b>	<b>773</b>

<b>Function Index</b>	<b>805</b>
<b>Include File Index</b>	<b>861</b>
<b>Constants and Defined Macros Index</b>	<b>867</b>





---

# Chapter 1 Introduction

- ▼ State of the Manual
- ▼ About the CUBIT Project
  - ▼ About this Manual
- ▼ Problem Reports and Enhancement Requests
- ▼ Additional Information

*CUBIT is the Sandia National Laboratories automated mesh generation environment. With CUBIT, you can create a finite element mesh using any combination of techniques including paving, mapping, and sweeping. CUBIT also features boundary layer meshing specifically designed for fluid flow problems. CUBIT is designed to reduce the time required to create all-quadrilateral and all-hexahedral meshes. CUBIT is intended to be both a production meshing tool and a research platform for meshing algorithm development. Current research meshing techniques in CUBIT include plastering and whisker-weaving which will provide automatic hexahedral meshing of arbitrary three-dimensional bodies.*

---

Note: All Bibliography references in this document reference the list in “Additional Information” on page 17 rather than a separate “References” section at the end of the manual.

---

## ▼ State of the Manual

This is the initial implementation of this manual. It is currently in a draft form and was last updated on March 25, 1997. It is not yet complete or finished, but its current state is complete enough that we believe it is useful to distribute it at this time.

## **Class Documentation**

Approximately one-half of the classes have been upgraded to the new documentation standards specified in Chapter 2, "Recommended Practices for CUBIT Code Development" on page 21. Of these classes, most have simply had header comments added, while only a few have had extended documentation/comments added. Although this sounds discouraging, there is actually a lot of useful information present in the manual. The extensive cross-referencing, both within the classes and in the indices at the end of the manual make it much easier to understand the flow of the CUBIT code.

## **Overview Documentation**

Scattered throughout the class documentation in this manual are chapters which attempt to provide an overview of the classes following that chapter. These chapters are intended to describe some of the unifying concepts within the CUBIT source code. The content of these chapters is admittedly minimally useful at this time. We will be expanding and improving this documentation extensively in the future.

## **▼ About the CUBIT Project**

The purpose of the CUBIT project is twofold:

- Provide a production two- and three-dimensional meshing environment containing powerful meshing algorithms that require minimal input to produce a complete finite element model, and
- Provide an easily extensible baseline platform for meshing algorithm research and development.

## **Production Tool Capabilities**

A summary of the major CUBIT production tool capabilities are listed in this section. For more information see the CUBIT Users Manual<sup>1</sup>.

- CUBIT provides both a graphical user interface and a conventional command line interface.
- CUBIT provides basic geometry creation and modification capabilities within the program and can read an external solid model file created by any of the several environments that support the ACIS solid model format.

- CUBIT provides automated and manual feature consolidation routines to identify and remove redundant geometry entities.
- CUBIT provides a highly automated, but flexible, mesh generation environment. Meshing is controlled through scheme choice and interval, or meshing density, selection. Curve meshing schemes include equally spaced or biased discretization. Surface meshing schemes include mapping transformations and paving. Volume meshing schemes include mapping transformations, mesh sweeping or projection, and automatic plastering of certain geometries.
- CUBIT supports several finite element types including 4, 8, and 9 node quadrilaterals; 4, 8, and 9 node shells; and 8, 20, and 27 node hexahedral elements.
- CUBIT writes the generated mesh and mesh attributes to a user-specified file in the EXODUS II<sup>18</sup> file format.
- CUBIT maintains a journal file of the CUBIT commands used during the mesh generation session. This file can be used in subsequent CUBIT modeling sessions.
- CUBIT can display wireframe and shaded representations of geometric or mesh entities, and hiddenline views of mesh entities. PostScript files can be created for any display.
- CUBIT currently executes on the Unix operating system on Sun, Hewlett Packard, and SGI workstations.

## **Research Tool Capabilities**

CUBIT is written using the C++ Programming Language. The main features of CUBIT are:

- CUBIT provides a large collection of basic data types useful for algorithm development including vectors, points, and linked-lists. (See “[DLLList Implementation](#)” on page 49. and “[Miscellaneous Classes and Constants](#)” on page 61)
- CUBIT provides basic mesh generation capabilities for Line, Surface, and Volume meshing that can be used as a basis for, or in conjunction with, new meshing algorithms. (See “[MeshTool Overview](#)” on page 45.)
- CUBIT provides an interface to the ACIS solid modeling database. (See “[AcisGeometryEngine Description](#)” on page 55.)
- CUBIT provides an interface to a graphical system which can display wireframe and shaded representations of geometric or mesh entities, and

hiddenline views of mesh entities. Graphical display of new entities can be easily added. (See “Overview of Graphics-Related Classes” on page 65.)

- CUBIT provides extensible classes for several geometric entities including vertices, wires, edges, faces, lumps, and bodies. The interface to the solid modeler is also described here. (See “Overview of CubitObject Classes” on page 33.)
- CUBIT provides extensible classes for several mesh-related entities including nodes, points, edges, faces, surfaces, hexes, and boundary conditions. (See “Overview of CubitObject Classes” on page 33.)
- CUBIT provides easily extensible command parsing utilities. (See “Overview of the User Interface and Graphics Related Classes” on page 57.)

## ▼ About this Manual

This manual documents the interface to the classes in the CUBIT source code. It is intended to be the internal documentation of the CUBIT source code for code developers and meshing algorithm researchers. It is not a user’s manual which is available separately in the document entitled “CUBIT Mesh Generation Program, Volume 1: Users Reference Manual<sup>1</sup>“. Nor does it contain an in-depth description of all of the algorithms used within the source code. That information is available separately either by perusing the source code, or in separately published documents (See “Additional Information” on page 17.)

The information in this manual is divided into several sections which roughly follow the distribution of the classes within the class hierarchy. Each of the major sections contains a chapter providing an overview description of the classes within that section. The major sections are:

- “Overview of CubitObject Classes” on page 33 contains the geometric (vertices, wires, edges, faces, lumps, and bodies), mesh (nodes, points, edges, faces, sheets, hexes, and whiskers), and related classes.
- “Overview of Tool Classes” on page 43 contains the classes related to drawing, geometry, and meshing manipulation or execution.
- “MeshTool Overview” on page 45 provides further detail on the edge, surface, and volume meshing tools.

- “DLList Implementation” on page 49 provides a detailed description of CUBIT’s implementation of doubly-linked lists which are used extensively within CUBIT.
- “Overview of the User Interface and Graphics Related Classes” on page 57 provides an overview of the user interface, command parsing, and graphics-related classes and the details of adding new commands.
- “Miscellaneous Classes and Constants” on page 61 provides an overview of the remainder of the classes that do not fit in the previous sections. These classes include some global variables, EXODUS II related classes, and some data types.

## Cross Referencing

This manual has been designed to make it easy to locate information about a specific class and related classes through extensive cross-referencing of classes, functions within the classes, include files, constants, and defined macros. The cross-references in this document are:

- A “hierarchical” table of contents is provided on page xi at the beginning of the manual. This table of contents indicates the derivation of each class and gives the page number for each class within the document.
- Similar information is presented in the “Class Index” on page 73 which provides an alphabetical listing of each class which indicates the location of the class declaration description, the name and location of the base class (if any) for this class, the names and locations of all classes which are derived from this class (if any), and the names and locations of any classes in which this class appears.
- The “Function Index” on page 105 is an alphabetical index of all functions declared in the CUBIT classes. In addition to showing the location of the function declaration, this index also specifies the type of the function (public, protected, private, friend, constructor, destructor, static, and/or virtual). Multiple declarations of the same function with different types are listed separately.
- “Constants and Defined Macros Index” on page 149 provides an index of all constants and defined macros.
- “Include File Index” on page 155 provides an index of the included files.
- Within each class description, each reference to another class has a cross reference to the location of that classes description.

## Available Forms of this Manual

This manual is available in three forms:

- A hardcopy SAND report,
- A hypertext FrameMaker file which can be viewed online using either FrameMaker or FrameView, and
- A Hypertext Markup Language (HTML) formatted document which can be viewed online using one of the several available HTML viewers including Mosaic<sup>28</sup> and Lynx<sup>29</sup>.

The two hypertext forms of this document are updated frequently as modifications are made to the CUBIT source code, so they will always contain the most up-to-date and accurate information. The FrameMaker source for the SAND report will also be updated simultaneously; however, it will not be published that frequently. Contact the authors for information on obtaining the manual in the various formats.

## Documentation Generation

The class description portions of this manual are generated automatically by utilities that extract the information directly from the specially-formatted class declaration (header) files. The class formatting requirements are summarized in the following chapter (“Recommended Practices for CUBIT Code Development” on page 21). The full description of the formatting requirements along with a description of the utilities used to generate the documentation is found in Reference 20, “*Class2frame and Class2html: Documentation Utilities for C++ Classes*.”

## ▼ Problem Reports and Enhancement Requests

The CUBIT project is using GNATS<sup>27</sup>, the GNU Problem Report Management System, for tracking questions, problem reports, and enhancement requests. The GNATS system is designed to allow users and developers who have problems with the CUBIT code, or who have requests for new features to be added to the CUBIT code to submit reports of these problems or requests to the internal Sandia Laboratories CUBIT developers. The GNATS system provides a program called *send-pr* which can be used to submit problem reports and enhancement requests in a standard, defined format which can be read by an

electronically managed database which automatically notifies responsible parties of the existence of the problem. It also provides a mechanism for keeping everyone involved in the problem report informed of the current state of the problem.

In general, you can use any editor and mailer to submit valid Problem Reports (PR), as long as the format required by GNATS is preserved. However, *send-pr* automates the process and ensures that certain fields necessary for automatic processing are present. The use of *send-pr* is strongly recommended for all initial problem-oriented correspondence with the CUBIT developers. The user documentation and source code for the GNATS system are available in the CUBIT distribution directory. If you have problems using the system contact Greg Sjaardema ([gdsjaar@sandia.gov](mailto:gdsjaar@sandia.gov)).

---

**Note:** The existence and recommended use of an electronic bug reporting mechanism is not an attempt by the CUBIT developers to ignore and or discourage face-to-face discussion of problems with or enhancements to the CUBIT code with users and developers. The use of GNATS is intended to help the developers manage, prioritize, and track the tasks required to produce a usable state of the art mesh generation package.

---

## ▼ Additional Information

In addition to this manual, additional technical reports describing the meshing algorithms and file formats may also be helpful to code researchers and developers interested in meshing algorithm development in general and CUBIT development in particular. This list is by no means an exhaustive coverage of meshing algorithm research, rather it is intended to list information directly related to the coding and algorithms used in the CUBIT project.

### Meshing Related

1. CUBIT Mesh Generation Program, Volume 1: Users Reference Manual, Sandia National Laboratories, 1994 (in preparation).
2. T. D. Blacker and M. B. Stephenson, ‘Paving: a new approach to automated quadrilateral mesh generation’, SAND90-0249, Sandia National Laboratories, (1990).
3. M. B. Stephenson, S. A. Canann, and T. D. Blacker, ‘Plastering: a new approach to automated, 3D hexahedral mesh generation’, SAND89-2192, Sandia National Laboratories, (1992).

4. T. D. Blacker and R. J. Meyers, "Seams and Wedges in Plastering: A 3-D Hexahedral Mesh Generation Algorithm," *Engineering with Computers*, **Volume 9**, 83-93 (1993).
5. T. D. Blacker, *FASTQ Users Manual Version 1.2*, SAND88-1326, Sandia National Laboratories, (1988).
6. W. A. Cook and W. R. Oakes, 'Mapping methods for generating three-dimensional meshes', *Comp. mech. eng.*, **Volume 1**, 67-72 (1982).
7. R. E. Jones, *QMESH: A Self-Organizing Mesh Generation Program*, SLA - 73 - 1088, Sandia National Laboratories, (1974).
8. R. E. Tipton, 'Grid Optimization by Equipotential Relaxation', unpublished, Lawrence Livermore National Laboratory, (1990).
9. A. P. Gilkey and G. D. Sjaardema, *GEN3D: A GENESIS Database 2D to 3D Transformation Program*, SAND89-0485, Sandia National Laboratories, (1989).
10. G. D. Sjaardema, *GREPOS: A GENESIS Database Repositioning Program*, SAND90-0566, Sandia National Laboratories, (1990).
11. G. D. Sjaardema, *GJOIN: A Program for Merging Two or More GENESIS Databases*, SAND92-2290, Sandia National Laboratories, (1992).

## Graphics Related

12. Open Software Foundation, Inc., *OSF/Motif™ Programmer's Guide Revision 1.1*, PTR Prentice Hall, Englewood Cliffs, New Jersey, (1991).
13. Open Software Foundation, Inc., *Application Environment Specification (AES) User Environment Volume Revision B*, Prentice Hall, Englewood Cliffs, New Jersey, (1991).
14. The Engineering Computer Graphics Laboratory, CQUEL.BYU, Brigham Young University, (1992).

## ACIS® Solid Modeling

15. Spatial Technology, Inc., *ACIS® Test Harness Application Guide Version 1.5*, Spatial Technology, Inc., Applied Geometry, Inc., and Three-Space, Ltd., (1993).
16. Spatial Technology, Inc., *ACIS® Geometric Modeler API Manual Version 1.5*, Spatial Technology, Inc., Applied Geometry, Inc., and Three-Space, Ltd., (1993).
17. Spatial Technology, Inc., *ACIS® Geometric Modeler Programmers Manual Version 1.5*, Spatial Technology, Inc., Applied Geometry, Inc., and Three-Space, Ltd., (1993).

## **EXODUS II File Format**

18. L. A. Schoof, *EXODUS II Application Programming Interface*, Sandia National Laboratories, (In Preparation).

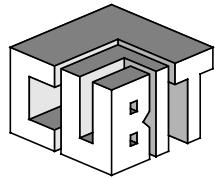
## **Programming Related**

19. W. J. Bohnhoff, “*Recommended Practices for CUBIT Code Development*,” internal memo, Sandia National Laboratories, February 1994.
20. G. D. Sjaardema, “*Class2frame and Class2html: Documentation Utilities for C++ Classes*,” internal memo, Sandia National Laboratories, February 1994.
21. J.A. Schutt, “*Suggested Style Guide for C++ Programming*”, Internal Memo, Sandia National Laboratories, October 1991.
22. Per Cederqvist, “*Version Management with CVS*,” *Release 0.8 for CVS 1.3-s2/1.3.1*,” March 15, 1993. Available from anonymous ftp to think.com in /pub/cvs/doc/cvs-texi-0.08.tar.gz.
23. M. Henricson and E. Nyquist, *Programming in C++, Rules and Recommendations*, Ellemtel Telecommunication Systems Laboratories, Document Number M 90 0118 Uen, Älvsjö, Sweden.
24. L. W. Cannon, R. A. Elliott, L. W. Kirchhoff, et.al, “*Recommended C Style and Coding Standards*,” Bell Labs, Revision 6.0, June 25, 1990.
25. T. Cargill, *C++ Programming Style*, Addison-Wesley Professional Computing Series, 1992.
26. D. F. Rogers, *Procedural Elements for Computer Graphics*, McGraw-Hill Book Company, 1985.
27. J. M. Osier, *Keeping Track, Managing Messages with GNATS, The GNU Problem Report Management System*, Users manual for GNATS Version 3.2, Cygnus Support, October 1993.

## **Other**

28. Mosaic is a World Wide Web Hypertext Markup Language client program which is available for Unix, Macintosh, and Microsoft Windows based computers. It is available from the National Center for Supercomputing Applications (NCSA). Mosaic is available from [ftp.ncsa.uiuc.edu:/pub/Web/Mosaic](ftp://ftp.ncsa.uiuc.edu:/pub/Web/Mosaic)
29. Lynx is a vt100-based distributed hypertext browser with full World Wide Web capabilities. Lynx is available from [ftp2.cc.ukans.edu:pub/lynx](ftp://ftp2.cc.ukans.edu:pub/lynx)





---

## Chapter 2 Recommended Practices for CUBIT Code Development

- ▼ Style Conventions
- ▼ Filenaming Conventions
- ▼ Documentation Comment Conventions
- ▼ Options for Producing Class Documentation
- ▼ Summary
- ▼ References

*The purpose of this chapter is to communicate recommended practices to other CUBIT developers so that the team adheres to a consistency which will lead to improved readability of the C++ code. Many of the adopted conventions come directly from Jim Schutt's Style Guide[1] while others resulted from discussions at project meetings. In addition to establishing recommendations, the guidelines provided in this chapter should be useful in producing the necessary documentation for our code development effort.*

### ▼ Style Conventions

The guiding theme of this style guide is to enable a programmer to discern at a glance what the type and scope of a variable or function is. The style adopted by CUBIT is as follows:

- Class names should be composed of two or more descriptive words, with the first character of each word capitalized:

*Example:*    `class ClassName;`

- Class member variables should be composed of two or more descriptive words, with the first character of the second and succeeding words capitalized:

*Example:*    `float classMemberVariable;`

- Temporary (i.e. automatic) variables are lower case, with underscores separating words in a multiple word temporary variable:

*Example:*    int temporary\_variable;

- Constants (i.e. parameters) are upper case, with underscores separating words:

*Example:*    const double CONSTANT\_VALUE;

- Function names are lower case, with underscores separating words:

*Example:*    int function\_name();

-There is no need to distinguish between member and non-member functions by style, as this distinction is usually clear by context.

-This style convention arose from the desire to have a member function which returns the value of a private member variable:

```
Example:    int memberVariable;
              int member_variable() {
                  return memberVariable;
              }
```

- Use typedefs in a single header file to redefine basic types, so that program precision can be changed by changing a few lines of typedefs rather than many lines of code. Use the same style as for class names:

*Example:*    typedef int Integer;
 typedef float Real;

- “use lots of white space to keep code readable!”. However, some discretion should be used in following this recommendation. For example, the use of white space on either side of the dot (.) or arrow (->) operators when accessing class members is discouraged.

## ▼ Filenaming Conventions

In addition to the style outlined above, the following filenaming conventions have been established for the CUBIT project.

- File names for C++ classes should be identical to the class name defined by that file. The type of file is determined by one of the four filename extensions listed below:

**.hpp** A class header file ends in the suffix .hpp. The header file provides the class declaration. This file does not contain code for implementing the methods,

except for the case of inline functions. Inline functions are to be placed at the bottom of the file with the keyword `inline` preceding the function name.

**.cc** A class implementation file ends in the suffix `.cc`. An implementation file contains the definitions of the members of the class.

**.h** A header file ends in the suffix `.h`. The header file contains information usually associated with procedures. Defined constants, data structures and function prototypes are typical elements of this file.

**.c** A procedure file ends in the suffix `.c`. The procedure file contains the actual procedures.

## ▼ Documentation Comment Conventions

The `class2frame` and `class2html`[2] programs will be used to extract documenting information directly from the C++ source code. All comments beginning ‘`//-`’ followed by a space or tab will be extracted for documentation. The hyphen is used to distinguish such a comment from a normal C++ comment. One of several keywords follows the ‘`//-`’ sequence to indicate the type of information included in the comment.

- The **Class** keyword: (required)

The *Class* of a file describes the object defined in the file. In the example below, this file refers to the definition of an object called `SweepTool`.

*Syntax:*      `//- Class: <ClassName>`

*Example:*     `//- Class: SweepTool`

- The **Description** keyword: (required)

The *Description* keyword signals the beginning of the text that states the contents and purpose of the file.

*Syntax:*      `//- Description: <text>`  
                  `//- <continuation text>`  
                  `//- <continuation text>`  
                  `...`

*Example:*     `//- Description: This class creates a mesh on a 3D`  
                  `//- lump. The elements are created by projecting`  
                  `//- them in successive layers from the sourceFace`  
                  `//- to the targetFace. The sourceFace and target-`  
                  `//- Face must be topologically similar.`  
                  `//- The RefFaces linking the sourceFace and`  
                  `//- targetFace must be meshable using the`  
                  `//- Mapping scheme.`

- The **Assumptions** keyword: (optional)

The *Assumptions* keyword signals the beginning of the text that describes any assumptions used in the development of the code.

*Syntax:*        // - Assumptions: <text>  
                   // - <continuation text>  
                   ...

*Example:*      // - Assumptions: All faces have 4 nodes and edges,  
                   // - each edge of source and target face linked by  
                   // - single face which is meshable using  
                   // - the mapping scheme.

- The **Owner** keyword: (required)

The *Owner* keyword precedes the code owner's name.

*Syntax:*        // - Owner: <Owner's Name>

*Example:*      // - Owner: Greg Sjaardema

- The **Checked By** keywords: (required)

The *Checked by* keywords precede the reviewer's name. The date that the file was last reviewed follows the name.

*Syntax:*        // - Checked by: <Reviewer's Name>, <date checked>

*Example:*      // - Checked by: Randy Lober, 12/13/93

- The **Version** keyword: (optional)

The *Version* keyword is used to document the RCS/CVS revision information. The text following the keyword is *automatically* inserted by RCS.

*Example:*      // - Version: \$Id:\$

- Descriptive comments for methods:

Comments describing a member function's purpose should be contained in the .hpp file for the class. One or more lines of descriptive text may be used. The descriptive text should be placed immediately *after* the prototype declaration of the method. Information describing the arguments or the return value of the method may be documented here.

*Syntax:*        void member\_function();  
                   // - <Descriptive text>  
                   // - <continuation text>  
                   ...

*Example:*      int create\_mesh();  
                   // - Initiates creation of projected mesh.  
                   // - Returns {CUBIT\_TRUE} if successful,  
                   // - Returns {CUBIT\_FALSE} if error meshing.

## ▼ Options for Producing Class Documentation

A documentation utility has been written to aid in the documentation of C++ classes. The utilities make it easier to maintain current documentation of C++ classes by extracting documentation information directly from the C++ source code. The two main utilities are *class2frame* which creates a FrameMaker documentation file, and *class2html* which creates an Hypertext Markup Language file. In addition to the parsing the required keyword comments listed above, these utilities recognize several other formatting flags which can be used to improve the appearance of the output documentation. A few of these are described below, the full documentation of these utilities is found in Reference ??, *Class2frame and Class2html: Documentation Utilities for C++ Classes (Initial Release)*.

- Font changes *within* comments:

@b{Text}: Outputs Text in a **bold** font,  
@i{Text}: Outputs Text in an *italic* font,  
@s{Text}: Outputs Τεξτ in a σψυβολ (symbol) font,  
@c{Text}: Outputs Text in a fixed-width Courier font,  
{Text}: Same as @c{Text}.

- Normal comments are denoted by ‘// -’; the comment text will appear in the main section of the resulting FrameMaker document.
- Comments that begin with ‘// @’ will appear near the end of the document in a section entitled ***Additional Comments***.
- Comments that begin with ‘//’ are *not* output to the FrameMaker file.
- End of line comments that are on the same line with some source code can use ‘// -’ and will be output in an italic font on the same line as the code that they describe.
- A comment line that ends with ‘\$’ will cause a line break in the document at that point. This is needed to format, for example, a fixed width table:

*Example:*      // - {1     5}\$  
                      // - { 2   4 }\$  
                      // - {   3    }\$  
                      // - { 2   4 }\$  
                      // - {1     5}\$

Will appear in the output just as:

*Example:*      1     5  
                      2   4

3  
2 4  
1 5

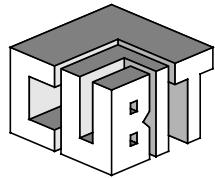
- An alternative approach to format the comment in a verbatim mode as shown above is to use a ‘//+’ at the beginning of the line. This is cleaner than using the ‘// - { . . . }\$’ sequence for fixed format comments.
- Comments that begin with ‘//@+’ will appear in a verbatim mode in the *Additional Comments* section.

## ▼ Summary

This chapter serves as a reference document for some basic principles that should lead to more consistent code development. This document can be updated if any developer has additional suggestions that would enhance these ideas.

## ▼ References

1. J.A. Schutt, “*Suggested Style Guide for C++ Programming*”, Internal Memo, Sandia National Laboratories, October 1991.
2. G.D. Sjaardema, “*Class2frame and Class2html: Documentation Utilities for C++ Classes*”, Internal Memo, Sandia National Laboratories, February 1994.



---

# Chapter 3 CUBIT Code Features

▼ Internal Message Reporting

▼ AutoTester

*The purpose of this chapter is to describe features of CUBIT available to CUBIT code developers. These features either provide a consistent interface for common functions like the reporting of errors from classes within CUBIT, or provide tools to aid in CUBIT code development.*

## ▼ Internal Message Reporting

Messages not related to input processing in CUBIT should be handled with the *CubitObject::internal\_error* function. This function controls the printing of debug, information, warning and error messages, with the output of each type of message controlled with a user-settable toggle.

To use the *internal\_error* function within another function, that other function must either be derived from *CubitObject* or have **CubitObject.hpp** included somewhere in the module. The *internal\_error* function is declared static, so if a function is not derived from *CubitObject*, an absolute reference must be made (*CubitObject::internal\_error(...)*). The declaration for this function is:

```
static void internal_error(int message_type,int stop_flag,  
char *format, ... );
```

The arguments are as follows:

- *message\_type* is one of the following: `CUBIT_ERROR`, `CUBIT_INFO`, `CUBIT_WARNING` or `CUBIT_DEBUG_[1-10]`.

The printing of each type of message is controlled with a toggle which can be set by the user. For example, the user can choose whether or not `CUBIT_WARNING` messages are printed. There are multiple debug flags, corresponding to multiple classes of debug messages. These flags are allocated in the beginning of file `CubitObject.cc` and in the Developer's Manual at the end of this section. `CUBIT_ERROR` messages are always printed, and have no controlling toggle.

- `stop_flag`

This argument is either `CUBIT_PROCEED` or `CUBIT_TERMINATE` (`internal_error` will call `exit(1)` if the latter is used).

- `format, ...`

These are arguments that would normally be passed to a printf-type function, where the elipses (...) denote a variable number and type of arguments. For example, the format can contain letters to be printed plus format characters corresponding to the data:

```
internal_error(message_type, stop_flag,
    "This is warning no. %d", (int) warningno)
```

These arguments are passed directly to a printf-type function, and so all legal format strings are allowed. There is only one addition to printf-type functionality provided by `internal_error`. That is, an end-line is passed to `cout` after the message has been processed (this was done for historical reasons). So, if a newline is passed in the format string, there will be an additional blank line printed after the message after the call to `internal_error`.

Also available are functions which return the values of flags. The following functions are available:

```
static int CubitObject::debug_flag(const int index);
static int CubitObject::info_flag();
static int CubitObject::warning_flag();
```

where the `index` argument in `debug_flag` is the debug flag number.

To change the number of debug flags available in CUBIT (the current maximum is 10), the `CUBIT_DEBUG_FLAGS` macro in **CubitObject.hpp** must be increased and values must be assigned to the new flags (`CUBIT_DEBUG_11`, `CUBIT_DEBUG_12`, etc). A record of debug flag allocation is kept in **CubitObject.cc** (it is kept there instead of in **CubitObject.hpp** because of the many compilation dependencies on the latter file), and is repeated here for reference:

```

// CUBIT_DEBUG_FLAG "Ownership":
//
// CUBIT_DEBUG_1: User Interface related. If flag is enabled,
//   filenames being used for input will be echoed. Also,
//   each input line will be echoed prior to being parsed.
// CUBIT_DEBUG_2: Whisker weaving information
// CUBIT_DEBUG_3: Timing information for 3D Meshing routines.
// CUBIT_DEBUG_4: Testing of video generation - if on then
//   video specific drawing is enabled and related debug
//   statements will be printed
// CUBIT_DEBUG_5: Whisker weaving crossing elimination
//   information
// CUBIT_DEBUG_6:
// CUBIT_DEBUG_7:
// CUBIT_DEBUG_8:
// CUBIT_DEBUG_9:
// CUBIT_DEBUG_10:

```

## ▼ AutoTester

*autoTester* is a script designed to perform tests of the CUBIT executable, compare the test output to standards, and then report any differences found in output. One of the major uses of *autoTester* is to run daily or periodic automatic tests of the CUBIT executable(s) undergoing changes by a number of developers. It can also be used locally by developers to test modifications before these changes are committed to the CUBIT repository or before they are incorporated in the CUBIT executable.

### Installation

*autoTester* is a script which executes CUBIT for a specified subset of test problems and compares results against standard versions of output files. In a standard configuration, testing is done under a directory, referred to here as \$(AUTOTESTER). This directory contains the following subdirectories:

- \$(AUTOTESTER)/util: location of *autoTester* and other scripts
- \$(AUTOTESTER)/test1, \$(AUTOTESTER)/test2, ...: subdirectories containing individual or multiple test problems.

Before using *autoTester* to test changes to CUBIT, a set of standards must be established. This is done by running the *buildstd* script while in the `$(AUTOTESTER)` directory:

```
buildstd all
```

This will build standard files in a specified set of test subdirectories (see the *buildstd* script for a list of standard tests included with *autoTester*). Note that the executable used to build the standard files is defined in *buildstd*.

*autoTester* uses the program *ncdump* to produce a text file containing mesh position and connectivity information. This program is part of the SEACAS code system and can be obtained from Sandia National Laboratories upon request.

### Running *autoTester* Manually

*autoTester* can be run manually, from any directory, to test the results of a particular executable for one of the test problems against the results from the standard executable. Several environment variables must be initialized before running *autoTester* manually:

- `CUBIT_EXEC` - the executable being tested against the standard
- `CUBIT_TEST` - the `$(AUTOTESTER)` directory referred to in the previous section
- `CUBIT_UTIL` - the `$(AUTOTESTER)/util` directory
- `CUBIT_OUT` - the directory in which results files are placed

After these environment variables are initialized, *autoTester* can be run on specific problems by entering the command

```
$(CUBIT_UTIL)/autoTester test1 test2 ...
```

where test1, test2, etc are the names of the tests. *autoTester* will run `$CUBIT_EXEC` for the specified tests in the current directory. The results are compared with those from the standard, storing any differences in `$CUBIT_OUT/result<date_and_time>`, where the current date and time are appended to the filename.

### Adding Standard Tests

Tests can be added to the standard tests in `$(AUTOTESTER)` using the following procedure.

- Create a subdirectory under `$(AUTOTESTER)`, giving it a descriptive name (referred to here as `$(SUBDIR)`).

- Place a journal file in the subdirectory with the name \$(SUBDIR)/\$(SUBDIR).test. This journal file should create an exodusII output file named \$(SUBDIR)/\$(SUBDIR).exoII, to facilitate comparison of the mesh created by the test problem. If a geometry description file is needed, this should be placed in the file \$(SUBDIR)/\$(SUBDIR).sat, or should be referenced with an absolute path name in the journal file.
- In the parent directory, run the buildstd script for that subdirectory:  

```
buildstd $(SUBDIR)
```

This script will execute CUBIT on the journal file in \$(SUBDIR) and create standard versions of the CUBIT standard output and an ncdump of the exodusII file produced and place them in files \$(SUBDIR)/\$(SUBDIR).std-out and \$(SUBDIR)/\$(SUBDIR).ncdump, respectively. These files are used to compare with subsequent test runs.

- Add this test name to the list of standard tests in *buildstd*, so that the test is built when ‘**buildstd all**’ is run.

*autoTester* has the capability of running several journal files in a single test subdirectory, as long as their filename extensions are all ‘.test’. This procedure is not recommended, however; the same capability can be achieved by using the ‘**reset**’ command within a single CUBIT journal file.

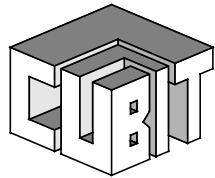
The tests that are currently part of the test suite and the CUBIT capabilities tested in each problem are shown in Table 1. This list is being expanded to incorporate tests for the new capabilities being added to CUBIT on an ongoing basis.

**Table 1. Test Problems and Features Tested in the CUBIT Test Suite.**

Examples	Geometry Features				Surface Meshing Features				Volume Meshing Features				
	Primitives	Booleans	Geometry Editing	Geometry Consolidation	Curve Bias	Curvature-based	Mapping	Paving	Triangle Tool	Boundary Layer Tool	Project	Translate/Rotate	Mapping
Internal Geometry	x	x					x				x		

**Table 1. Test Problems and Features Tested in the CUBIT Test Suite.**

Examples	Geometry Features				Surface Meshing Features				Volume Meshing Features			
	X	X	X	X			X	X		X	X	
Sphere Octant	X	X	X	X			X	X		X	X	
Box Beam				X		X						
Thunderbird							X					
Assembly Components				X			X			X		
Knee				X			X	X		X	X	X
Block Project						X				X		
Block Map						X					X	



---

## Chapter 4 Overview of CubitObject Classes

▼ Derived Classes

▼ The Model Class

*The CubitObject class is the base of the CUBIT class hierarchy for all of the Command-related, Geometry-related, and Mesh-related classes. It contains the enumerations that are used in its derived classes to maintain consistency; the globally accessible defines such as the CUBIT version, date, and ACIS version.*

An error reporting function is defined in the CubitObject class which should be used for all internal error checks. It is called with two arguments; the first argument is the error string that is printed to standard output, preceded by the string “ERROR:”, the second argument is an integer flag that determines whether the code should exit due to this error.

### ▼ Derived Classes

The classes derived from CubitObject are:

- **GeometryEntity** (See “Overview of GeometryEntity Classes” on page 35.) which is the base of the hierarchy for the geometry-related classes.
- **MeshEntity** (See “Overview of MeshEntity Classes” on page 39.) which is the base of the hierarchy for the mesh-related classes.
- **Tool** (See “Overview of Tool Classes” on page 43.) which is the base of the hierarchy for the classes related to the actual tools and algorithms used in the meshing process.
- **Commands, UserInterface** and other command-related classes (See “Overview of the User Interface and Graphics Related Classes” on page 57.). These classes are not in a subhierarchy like the previous classes, rather they all derive directly from CubitObject.
- **Model** is the overall “goal class” in the CUBIT system. This means that the entire purpose of the CUBIT system is to build a “Model” which consists of a geometry ready to be used as input to a finite element calculation.

## ▼ The Model Class

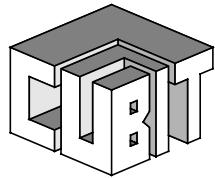
The Model class is essentially a collection of lists of pointers to all of the pieces that comprise the finite element model being created.

- Geometry lists contain pointers to all of the underlying solid model geometry are maintained (`RefBody`, `RefLump`, `RefFace`, `RefWire`, `RefEdge`, `RefVertex`),
- These lists can be used to get the mesh entities assigned to the geometric entities.
- Boundary condition lists contain pointers to `SideSets` and `NodeSets` which are the boundary conditions supported in the Exodus II binary file output format<sup>18</sup>.

The class contains functions which are used to inquire about the state of the current model, determine ownership of geometric quantities, set and retrieve identification numbers of geometric and mesh quantities, add and remove geometric entities to and from the model, and delete the entire mesh<sup>1</sup>.

---

1. At the current time, only the entire mesh can be deleted from the model. In a future release, portions of the mesh will be able to be deleted on a geometry entity by entity basis.



---

# Chapter 5 Overview of GeometryEntity Classes

## ▼ Geometry Definition

*This chapter describes the classes in CUBIT that are related to geometric entities. Definitions of geometric entities and the structure of the nonmanifold geometry represented by CUBIT are given.*

## ▼ Geometry Definition

All geometric entities that exist in the CUBIT environment are represented by a solid model. In two dimensional modeling systems, a wire frame representation is sufficient to provide a complete and unambiguous geometric definition. In three dimensions, only a solid model representation can guarantee complete and unambiguous geometry. All of the meshing tools available in CUBIT use the solid model as their basis when generating the discretized representation of the geometry, i.e. the mesh. ACIS® is the solid modeling engine currently used by CUBIT. However, CUBIT uses its own geometric overlay to represent a non-manifold cellular topology for meshing.

### Geometric Topology

Topology refers to the manner of connecting the geometric entities within the solid model. Within CUBIT, the geometric entities are RefBody, RefLump, RefFace, RefEdge, RefWire, and RefVertex. These correspond to the ACIS ENTITY classes BODY, LUMP, FACE, EDGE, WIRE, and VERTEX, respectively. See Section 5 of *ACIS® Geometric Modeler Programmers Manual*<sup>17</sup> for details on the corresponding ACIS classes.

- **RefBody**

RefBodys are the highest level geometric entity in CUBIT. A RefBody can consist of zero or more RefLumps, RefFaces, and RefEdges. A RefBody corresponds to the ACIS BODY class.

- **RefLump**

**RefLumps** are volumetric regions and are always bounded by one or more **RefFaces**. For practical consideration, **RefLumps** will always be bounded by two or more **RefFaces**. CUBIT currently cannot mesh a **RefLump** bounded by only one **RefFace** (e.g. a sphere) since such a **RefFace** would have no bounding curves. A **RefLump** corresponds to the ACIS LUMP class.

- **RefFace**

A **RefFace** in cubit is a finite bounded portion of some geometric surface (finite or infinite) description. A set of **RefFaces** bound the volume in a **RefLump**. A **RefFace** is bounded by a set of **RefEdges**. **RefFaces** must have at least one bounding **RefEdge** to be meshed in CUBIT. A **RefFace** corresponds to the ACIS FACE class.

- **RefEdge**

A **RefEdge** is a line (not necessarily straight) which is bounded by at least one but not more than two **RefVertices**. An example of a single **RefVertex** **RefEdge** (both ends of the **RefEdge** meet the same vertex) might be the **RefEdge** which bounds an end cap **RefFace** of a cylinder. A **RefEdge** is used to bound a **RefFace**. **RefEdges** may be generated independent of **RefFaces** and as such used to specify the geometry for a sequence of bar or beam elements. A **RefEdge** corresponds to the ACIS EDGE class.

- **RefVertex**

A **RefVertex** occupies a single point in space. A **RefVertex** is used to bound a **RefEdge** and/or to specify a specific location for a **CubitNode**. A **RefVertex** corresponds to the ACIS VERTEX class.

- **RefWire**

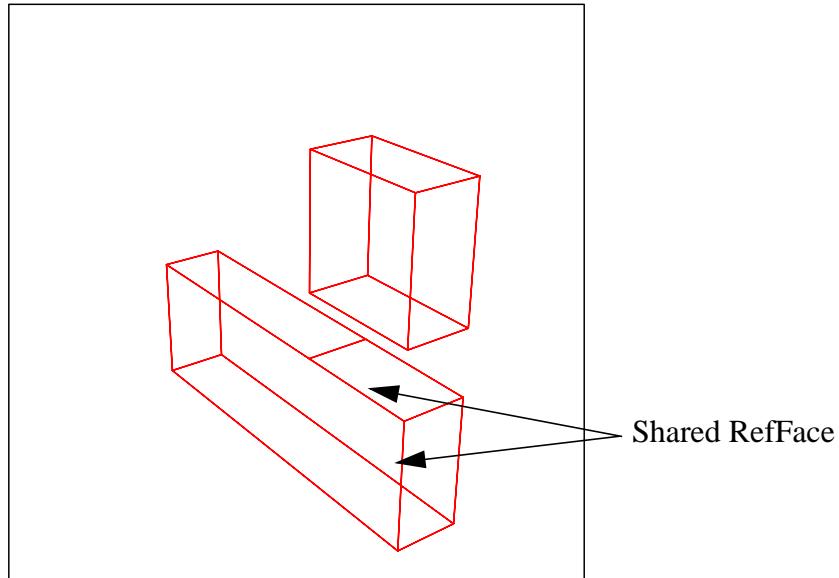
A **RefWire** is a set of connected **RefEdges**. A **RefWire** corresponds to the ACIS WIRE class.

## Cellular Topology

Cellular topology (or n-manifold topology) allows the connection of any number of **RefFaces** to **RefEdges**. A typical manifold model (or 2-manifold topology) only allows two **RefFaces** to be bounded by a single curve. Cellular topology, because of its more general nature, allows two adjacent **RefLumps** to share a common **RefFace** between them as shown in Figure 1. It also allows the formation of dangling **RefFaces** and **RefEdges**, which are geometric constructs useful in generating more complex meshes.

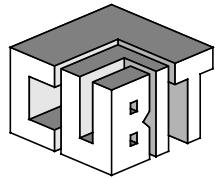
Cellular topology's advantage for mesh generation is that, when used properly, it eliminates the problem of equivalencing the mesh entities that are supposed

to be shared between adjacent geometric entities. For instance, the common surface shown in Figure 1.



*Figure 1* Illustration of Cellular Topology Face Sharing





---

# Chapter 6 Overview of MeshEntity Classes

▼ Definitions

▼ EdgeUses and FaceUses

▼ NodeHex Description

*This Chapter contains information about the classes relating to the Mesh Entities in CUBIT. The MeshEntity class is the base class for the typical finite element entities (nodes, one-, two-, and three-dimensional elements) and some algorithmic-specific mesh entities (Whisker-related classes and plastering related classes). The important EdgeUse and FaceUse classes are also derived from MeshEntity. These classes are used to maintain connectivity and avoid redundancy in the generated finite element mesh.*

---

Note: The information in this chapter has not been thoroughly debugged and checked for correctness. If there are any discrepancies between the information in this chapter and the information in the CUBIT source code, the source code is almost certainly correct.

---

## ▼ Definitions

The classes derived from the MeshEntity class include the typical finite element entities (nodes, edges, faces or surface elements, and hexahedral elements) plus some CUBIT-specific finite element entities (Whisker Faces, Chords, Hexes, and Sheets [Used in WhiskerWeaving]; CubitKnife; ProtoHex [Used in Plastering]; EdgeUse; and FaceUse).

### Finite Element Entities

Figure 2 shows a schematic of the various typical finite element entities and the corresponding CUBIT class names. The CubitHex shown in the figure is a FullHex; a NodeHex is similar except that CubitEdges and CubitFaces are not explicitly created. The NodeHex class was implemented to reduce the memory requirements and overhead of creating a hexahedral element when the functionality provided by the CubitEdges and CubitFaces was not required.

NOTE: This is a single CubitHex, not a RefLump

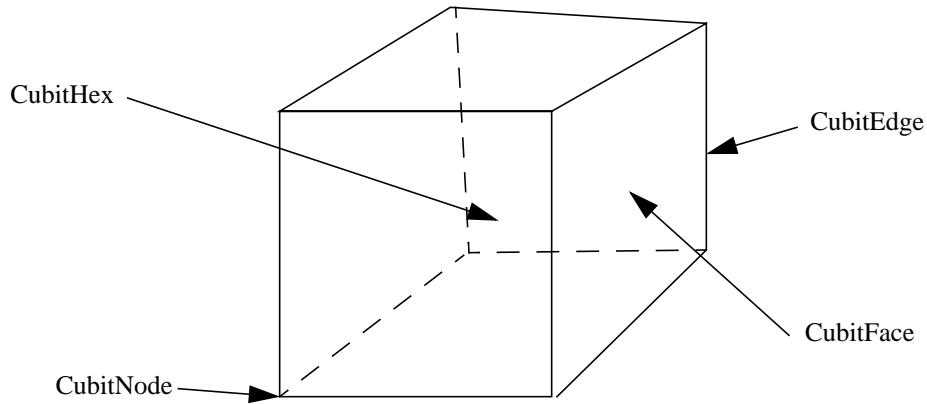


Figure 2 Typical Finite Element Mesh Entities and the Corresponding CUBIT Classes

## Algorithmic-Specific Mesh Entities

---

Note: This section is currently incomplete. WhiskerWeaving and Plastering related classes will be discussed here.

---

## ▼ EdgeUses and FaceUses

When multiple FullHexes are created, the CubitNodes, CubitFaces and CubitEdges that are created may be shared by multiple FullHexes. The volume meshing algorithms typically work from the CubitNodes, so they only create a CubitNode when a new one is needed so the problem with duplicate nodes is avoided. However, keeping track of the CubitEdges and CubitFaces is complicated enough that two classes EdgeUse and FaceUse were implemented to avoid the problem of creating duplicate CubitEdges and CubitFaces. When a FullHex is created from eight CubitNodes, the FullHex proceeds a face at a time to “create” a new FaceUse. The FaceUse constructor cycles through each of the four CubitEdges of the requested CubitFace and determines if it is an already existing CubitEdge or if a new CubitEdge must be created<sup>1</sup>. Once all CubitEdges are located and/or created, the FaceUse

---

1. Actually, the CubitEdge creation process occurs before the CubitFace process. This oversight will be corrected in the next release of this manual.

constructor must determine whether to create a new **CubitFace**, or use an existing **CubitFace**.

Each **CubitEdge** maintains a list of a list of the **CubitFaces** that it is connected to. When determining whether to create a new **CubitFace**, the **CubitFace\* FaceUse::find\_face()** function cycles through each **CubitEdge** on the requested new/old **CubitFace**. Each **CubitEdge** in turn searches its list of **CubitFaces** to see if one of them has the same edges as the tentative **CubitFace**. If a match is found, the pointer to that **CubitFace** is returned; otherwise, a new **CubitFace** is created, the **CubitEdges FaceUse** list is updated and the pointer to the new **CubitFace** is returned.

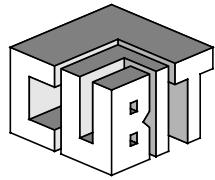
A similar process is performed when a new **CubitFace** is created: each **CubitEdge** is checked for existence and created only if it does not already exist.

Even though the search algorithms are efficient, the searching overhead along with the added storage for each **CubitFace** and **CubitEdge** add significantly to the execution time and memory usage in the generation of a **FullHex**. It is for this reason that the **NodeHex** class was implemented.

## ▼ NodeHex Description

The **NodeHex** class is a normal hexahedral element defined by the connectivity of eight **CubitNodes**. However, unlike the **FullHex**, it does not use or create **CubitEdges** and **CubitFaces** when it is constructed. Because of this, it can be created very quickly and efficiently; however, without the **CubitEdge** and **CubitFace** information, it can not be used in all the areas that a **FullHex** can be used. For example, the smoothing algorithms require **CubitEdges** and **CubitFaces** to locate neighboring elements and the boundaries of a **RefLump** also require this information for boundary condition applications. Therefore, **NodeHexes** are only used on the interior of **RefLumps** that are not smoothed in any way. Currently, **VolMapTool**, **RotateTool**, and **TranslateTool** are the only volume meshing tools that use **NodeHexes**. The execution time is typically reduced by a factor of five to ten when **NodeHexes** are used instead of **FullHexes**.



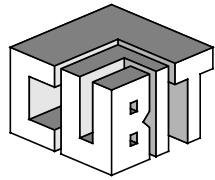


---

## Chapter 7 Overview of Tool Classes

▼ Incomplete at this time





---

# Chapter 8 MeshTool Overview

## ▼ Summary

---

Note: The state of this chapter is very preliminary. Much additional information is required here including descriptions of the smoothing algorithms and descriptions of the meshing algorithms. Information for adding new 2.5D meshing tools should also appear here.

---

## ▼ Summary

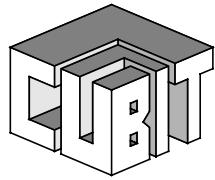
The following classes are derived from the `MeshTool` class:

- `EdgeMeshTool`: Create `CubitNodes` on a `RefEdge`. A new `CubitEdge` is also created corresponding to the `RefEdge`. The created `CubitNodes` and `CubitEdge` are owned by the `RefEdge`. Note: The `CubitNodes` at each end of the `RefEdge` are owned by the `RefVertex` at those points.
- `SurfMeshTool`: Creates `CubitNodes` and `CubitFaces` on a `RefFace`. The `RefEdges` are meshed first by the `EdgeMeshTool`, then the `CubitNodes`, `CubitEdges`, and `CubitFaces` are created on the interior of the `RefFace`. The `RefFace` owns all of the newly created mesh entities. The `SurfMeshTool` class consists primarily of member functions related to smoothing. The actual meshing functions are performed by the following classes:
  - `SurfMapTool`: Generate an all-quadrilateral mesh using the mapping transformation from Reference 6. The algorithm is a linear transformation in which all of the boundary edges can be curved and each is equally represented.
  - `Paver`: Generates an all-quadrilateral mesh using the Paving algorithm described in Reference 2.
  - `BoundaryLayerTool`: Inserts rows of elements around the boundary of a `RefFace` prior to meshing the interior. The aspect ratios of these elements can be carefully controlled. This capability is designed for fluid simulations involving a boundary layer. The interior can be meshed using either the `SurfMapTool` or the `Paver`.

- **TriangleTool:** The **TriangleTool** can be used to generate an all quadrilateral mesh on a triangular **RefFace** that contains 3 natural corners. For instance, **RefFaces** represented by the octants of a sphere are handled nicely by the **TriangleTool**. The algorithm requires that there be at least 6 intervals specified on the **CubitEdges** representing the perimeter of the **RefFace**.
- **VolMeshTool:** Creates **CubitNodes**, **CubitFaces**, and **FullHexes** or **NodeHexes** in a **RefLump**. The **RefFaces** are meshed first by the **SurfMeshTool**, then the **CubitNodes**, **CubitEdges**, **CubitFaces**, and **CubitHexes** (**FullHex** or **NodeHex**) are created on the interior of the **RefLump**. The **RefLump** owns all of the newly created mesh entities. The **VolMeshTool** class contains no member functions. The actual meshing functions are performed by the following classes:
  - **HexTool:** This class contains the member functions for smoothing a hexahedral mesh.
  - **Hexer:** This class uses the plastering algorithm to generate an all-hexahedral mesh. This class is currently at research status and is not ready for production usage.
  - **TwoHalfDimTool:** This is the base class for the 3D projection-type meshing tools. The **CubitHexes** are created by projecting them in successive layers from the source **CubitFace** to the target **CubitFace**. The class contains several member functions which are common to all/most projection-type algorithms which should make it relatively easy to add additional algorithms. The projection method depends on the projection meshing tool being used:
    - **RotateTool:** Creates both **NodeHexes** and **FullHexes** by projecting layers about a central axis. **FullHexes** are generated at the boundaries of the **RefLump** and **NodeHexes** are generated internal to the **RefLump**. No smoothing is performed.
    - **SweepTool:** This is the most general of the projection-type algorithms. **FullHexes** are created by projecting them in successive layers using only locally available geometric information. Smoothing is performed after each layer is generated.
    - **TranslateTool:** Creates both **NodeHexes** and **FullHexes** by projecting along a vector from the source **RefFace** to the target **RefFace**. **FullHexes** are generated at the boundaries of the **RefLump** and **NodeHexes** are generated internal to the **RefLump**. No smoothing is performed.

- **WhiskerWeaver**: An automatic hexahedral meshing scheme that is currently being researched. This method complements the plastering scheme in the **Hexer** class.
- **VolMapTool**: Generate an all-hexahedral mesh using the mapping transformation from Reference 6. The **CubitNode** locations are determined by mapping an unit cube in the volume represented by the **RefLump**. The algorithm is a linear transformation in which all of the boundary edges can be curved and each is equally represented.





---

# Chapter 9 DLLList Implementation

- ▼ Implementation Details
- ▼ Efficiency Details
- ▼ Creating a variant of a DLLList
- ▼ DLLList variants used in CUBIT

A *DLLList* is CUBIT's implementation of a doubly-linked, circular list. Lists are used extensively in almost every class within CUBIT. Due to their pervasiveness throughout CUBIT, it is important that CUBIT developers understand the underlying implementation details of DLLLists in order to write memory-efficient and time-efficient code. This chapter provides the details of the current DLLList implementation, instructions on how to use DLLLists efficiently, details on creating new DLLList variants, and a table detailing the various kinds of DLLLists currently used in CUBIT. The actual description of the DLLList Class is found in “*DLLList Description*” on page 205

## ▼ Implementation Details

---

Note: The DLLList implementation details described in this section are subject to change. The inner details of the class are described here only to make it easier to use the lists efficiently. Although the underlying data structure of a DLLList is an array, it is unwise to use a DLLList in a situation for which an Array class would be more appropriate.

---

The data in a DLLList are stored internally in a fixed-length array. The array stores pointers to the underlying class rather than the actual class data. This is important to realize since:

- Modifications of class data within a DLLList will modify the actual class data,
- Modification of class data external to a DLLList will modify the data within the DLLList, and
- Deletion of an element from a DLLList will not delete the class data

In addition to the array of pointers, a DLLList maintains an index into the array, the number of elements (pointers) in the array, the current size of the array, and the increment by which the array size is increased if the array fills up. Note that the number of elements in the list is not necessarily the same as the size of the array. Memory is allocated in a fixed size (increment) and items are added to or removed from the list while maintaining all unused space at the end of the array. If the list becomes full, a new array of size ‘current size + increment’ is allocated and the elements are copied from the old array into the new array. The memory allocated for the old array is then deleted.

Access to the array is circular, that is, the first element in the list follows the last element in the list and the last element in the list precedes the first element in the list.

Elements are stored contiguously within each list and all unused portions of the array are at the end of the array. If an element is deleted from a list, all elements from that point to the end of the list are shuffled down to fill the newly created void. If an element is inserted into a list, all elements from that point to the end of the list are shuffled up one position to open up a slot for the new element.

Several list access and list modification class member functions are provided for using DLLLists. These are fully described in “DLLList Description” on page 205. A few of these functions are described in the next section.

## ▼ Efficiency Details

The following short example will hopefully help motivate the need for understanding how to use DLLLists efficiently within CUBIT. Table 2 shows some execution timings for meshing a simple cube within CUBIT using the “project” meshing scheme. The first five rows of the table are for a mesh which is “X” intervals per side of the cube resulting in “X\*X\*X” total elements, the following seven lines of the table are for a mesh in which there are “X” layers of a 10\*10 mesh resulting in “10\*10\*X” total elements. The numbers in the table are the number of hexahedral elements generated per second of CPU time on one of our development computers.

- “Version B” refers to the baseline version of the code prior to any profiling and tuning,
- “Version T1” is the version of the code following some simple tuning of DLLList usages and minor algorithmic changes to modify the way in which the lists are accessed, and

- “Version T2” is the version of the code following some more involved algorithmic tuning and some more minor tuning of DLLList usages.

It is very evident from this data that the manner in which DLLLists are used within CUBIT can have a very measurable effect on the execution speed. The remainder of this section will detail some of the lessons that were learned about efficient DLLList usage during this tuning exercise.

**Table 2. Illustration of the effect of efficient versus inefficient use of DLLlists**

Intervals/ Side	Version B el/sec	Version T1 el/sec	Version T2 el/sec	Ratio: T1/B	Ratio T2/B
Cubic Mesh: Intervals x Intervals x Intervals					
10	426	524	730	1.23	1.71
15	336	900	1064	2.68	3.17
20	204	917	1176	4.50	5.77
25	132	893	1149	6.77	8.70
30	67	855	1099	12.7	16.4
Layered Mesh: 10 x 10 x Intervals					
10	433	658	730	1.52	1.69
20	459	813	926	1.77	2.02
30	395	877	1053	2.22	2.67
60	255	725	1163	2.84	4.56
120	144	763	1176	5.30	8.17
240	69	621	1075	9.00	15.6
360	46	524	952	11.4	20.7

## Guidelines for Efficient DLLList Usage

- Delete and insert items at the end of the list if at all possible.**

Since all data within a list is stored contiguously, whenever an element is deleted or inserted at any point other than at the end of the list, the elements between that point and the end of the list must be moved to fill or open up that location. The function `DLLList::last()` will position the index pointer at the end of the list. A significant portion of the time reductions shown in Table 2 was due to modifying the code so that it would append to the end of lists rather than inserting at the beginning.

- **Don't remove items unless necessary.**

Although it is very easy and somewhat elegant to iterate through a list using the following method:

```
while (DLLList.size() > 0) {
    item = DLLList.remove();
    ...perform some function
}
```

It is more efficient if the same function can be performed as:

```
for (int i=DLLList.size(); i > 0; i--) {
    item = DLLList.getItemAndStep();
    ... perform some function
}
```

If your algorithm requires that the list be empty at the end of this function, use the `DLLList::cleanOut()` function call to remove all items from the list.

- **Set the list growth increment to the known size of the list.**

The optional argument in the list constructor is the growth increment which is the amount of memory to be allocated for the list initially and whenever it needs to be grown. If the increment is not specified, the default value of 16 is used. A few other constants (defined in database.hpp) are provided that can serve as rough guidelines for “average” list sizes:

```
const int HEX_INCREMENT = 5000;
const int FACE_USE_INCREMENT = 6 * HEX_INCREMENT;
const int FACE_INCREMENT = 4 * HEX_INCREMENT;
const int NODE_INCREMENT = HEX_INCREMENT;
```

In general though, it is better to set the increment to the actual list size (if known) than to rely on the default value or the above constants. The growth increment can be modified after list creation by the `DLLList::set_increment(size)` command.

- **Use operator= and operator+= for assigning and appending lists.**

The assignment and compound assignment (= and +=) operator overload functions have been recently added to the DLLList classes to improve efficiency. Previously, it was necessary to copy or append lists using the code:

```
for (int i=from.size(); i > 0; i--)
    to.append() = from.getAndStep();
```

This is very efficient coding; however, the efficiency is improved, especially for long lists, by rewriting this as `to = from` or `to += from`. These are more efficient since error checking, list growth, and index manipulation/checking is done one time only rather for each insertion; and there is only one function call rather than `from.size()` function calls.

- **For some algorithms, it may be more efficient to use two lists,**  
Some of the smoothing functions in CUBIT use a DLLList as a first-in/first-out (FIFO) queue-like data structure in which an element is removed from the beginning of the list, operated upon, and as a result of these operations additional elements are appended onto the end of the list. This cycle continues until the list is empty. We were able to significantly speed up these functions by maintaining two lists rather than one. The functions now iterate through the first list (without removing items) and append the new elements to the end of a second list. After iterating through the first list, it is cleaned-out and the loop is begun again with the roles of the two lists reversed. This somewhat simple change eliminates all removals from the beginning of the list and does not significantly increase memory usage.
- **For some algorithms, it may be more efficient to create a new data type.**  
The previous example should, ideally, be solved by creating a Queue data structure in which there is no penalty for removing the first item in a list. In this case, it was more programmer-efficient to rework the algorithm; however, there are some algorithms within CUBIT where this is not possible. As time permits, new data types will be added to address these problems. The data types that may be added are previously mentioned Queues, and, for lists that need to be searched frequently, HashedLists or SortedLists.
- **Profile the code rather than making blind changes.**  
The output from the profiler was very informative during the tuning of CUBIT. The routines targeted for tuning prior to profiling the code were actually not responsible for the majority of the wasted time. The profiler also provided function call counts which pointed out a few inefficiencies in algorithms which were changed to reduce the number of calls. It is important during profiling to run several similar problems of different sizes to see the “growth” of an algorithm. Algorithms that run fine on small problems may become very inefficient on large problems. As an example, one function that used only ~1% of the execution time for the 10\*10\*10 mesh, used more than 80% of the execution time for the 30\*30\*30 problem.

## ▼ Creating a Variant of a DLLList

The DLLList class is written in terms of storing elements which are pointers to void. The DLLList class declaration file, DLLList.hpp, defines a macro called DLLListdeclare which is used to create list classes which are derived from a DLLList and can store pointers to any type. The DLLListdeclare macro is defined as:

```
#define DLLListdeclare(name,typePtr)
    class name : public DLLList
{
public:
    name (int size=0) : DLLList (size) {}
    void insert ( typePtr objPtr ) { insertLink ( (void *) objPtr ); }
    void append ( typePtr objPtr ) { appendLink ( (void *) objPtr ); }
    typePtr remove () { return (typePtr) cutLink(); }
    typePtr get () { return (typePtr) getItem(); }
    typePtr next () { return (typePtr) nextItem(); }
    typePtr next ( int n ) { return (typePtr) nextItem(n); }
    typePtr prev () { return (typePtr) prevItem(); }
    typePtr prev ( int n ) { return (typePtr) prevItem(n); }
    typePtr getAndStep () { return (typePtr) getItemAndStep (); }
    typePtr getAndBack () { return (typePtr) getItemAndBack (); }
    int moveTo ( typePtr objPtr ) { return (moveToItem ((void*)objPtr)); }
    typePtr changeTo ( typePtr objPtr )
    { return (typePtr) changeItemTo ( (void*) objPtr); }
}
```

The arguments to the macro are `name` which is the name of the list (for example, `DLCubitNodeList`), and `typePtr` which is a pointer to the type of data which will be stored in the list (for example `CubitNode*`). The convention used in CUBIT is that all lists created using this macro begin with “DL” and end with “List”. If possible, the remainder of the name should be the same as the class type stored in the list. A template for creating new lists following these guidelines is:

```
Put in FileName: DLclasstypeList
#ifndef DLclasstypeLIST_HPP
#define DLclasstypeLIST_HPP
#include "DLLList.hpp"
class classtype;
DLLListdeclare(DLclasstypeList,classtype*);
#endif
```

where `classtype` is replaced by the name of the class for which the list is created. The macro creates a class with several inline functions which convert pointers to the current type to/from the void pointers used in the base DLLList class.

This method will probably be replaced by a template DLLList class in the near future.

## ▼ DLList variants used in CUBIT

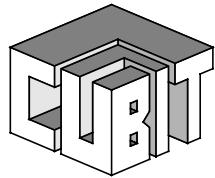
The following table documents the DLLists that are used in CUBIT. The name of each list, the class stored in the list, and a cross-reference to the definition of the list are listed in the table.

**Table 3. DLList Variants Used in CUBIT**

Name of List Class	Class Stored in List	Class Definition
DLBoundaryLayerEdgeList	BoundaryLayerEdge	See page 60.
DLBoundaryLayerFaceList	BoundaryLayerFace	See page 63.
DLBoundaryLayerList	BoundaryLayer	See page 95.
DLBoundaryList	DLCubitNodeList	See this table
DLBLEDgeLoopList	DLBoundaryLayerEdgeList	See this table
DLCOEDGEList	COEDGE	See ACIS Documentation
DLChordPointList	SheetChordPoint	See page 287.
DLCubitEdgeList	CubitEdge	See page 78.
DLCubitFaceList	CubitFace	See page 80.
DLCubitHexList	CubitHex	See page 87.
DLCubitNodeList	CubitNode	See page 74.
DLCubitSheetList	CubitSheet	See page 85.
DLCubitVectorList	CubitVector	See page 291.
DLDoubleList	Double	Built-in Type
DLEDGEList	EDGE	See ACIS Documentation
DLEdgeUseList	EdgeUse	See page 111.
DLElementBlockList	ElementBlock	See page 303.
DLFACEList	FACE	See ACIS Documentation
DLFaceUseList	FaceUse	See page 112.
DLGeometryEntityList	GeometryEntity	See page 30.
DLProtohexList	Protohex	See page 60.
DLLoopList	Loop	See page 302.

**Table 3. DLList Variants Used in CUBIT**

Name of List Class	Class Stored in List	Class Definition
DLMeshEntityList	MeshEntity	See page 72.
DLNodeLoopList	DLCubitNodeList	See this table
DLNodeSetList	NodeSet	See page 145.
DLProtoHexList	ProtoHex	See page 114.
DLRefBodyList	RefBody	See page 57.
DLRefEdgeList	RefEdge	See page 40.
DLRefFaceList	RefFace	See page 45.
DLRefLumpList	RefLump	See page 52.
DLRefVertexList	RefVertex	See page 35.
DLRefWireList	RefWire	See page 38.
DLSheetChordList	SheetChord	See page 260.
DLSideSetList	SideSet	See page 148.
DLVERTEXList	VERTEX	See ACIS Documentation
DLWhiskerChordList	WhiskerChord	See page 97.
DLWhiskerFaceList	WhiskerFace	See page 83.
DLWhiskerHexList	WhiskerHex	See page 102.
DLWhiskerSheetList	WhiskerSheet	See page 105.
DLbs3_curveList	bs3_curve	See ACIS Documentation
DLcurveList	curve	See ACIS Documentation
DLpositionList	position	See ACIS Documentation



---

## Chapter 10 Overview of the User Interface and Graphics Related Classes

▼ The Parser Class

▼ Abstract Command Classes

▼ Adding Code to Support a New Command

▼ Future Command Interface Development

*This section describes the mechanism by which a programmer can provide a command interface to the mesh tools being developed. For the most part, adding a new command to the CUBIT interface is as simple as writing a C function and associating a pointer to that function with a command string.*

----NOTE: This chapter describes the old parser, it is not up-to-date----

### ▼ The Parser Class

CUBIT has an internal Parser class that provides a lexical analyzer/parser generator of character based input. The Parser reads from either stdin or a string in memory. A command string is broken up into “tokens” that may be considered integer data, real data, or character (string) data. The tokens are stored in the C data structure, dataSet (defined in dataSet.h).

A CommandTable is defined and utilized by the Parser class to match a valid command string with a function pointer. The CommandTable is simply a one-to-one pairing of a string with a function pointer. As the Parser tokenizes the input string, the CommandTable is continuously searched until a match is found. Upon finding a match, the Parser calls the function through the function pointer and the tokens are passed in as an argument of the function call.

## ▼ Abstract Command Classes

The function pointers in the CommandTable are assigned to the address of global functions. Since CUBIT has been written in C++, the global functions may be encapsulated in classes. A variety of abstract classes (classes in which no objects may be instantiated) have been established to provide a logical grouping of similar C++ functions (e.g. FileCommands, GenesisCommands, WhiskerCommands, etc.). It is important to note that every function that provides a command interface has an identical prototype and is declared *static*. For example:

```
static void export_genesis(dataSet parsed_data);
```

declares the function that exports the Genesis database to a file (the filename is contained within the `parsed_data` passed as an argument). These functions should not be considered as true class members, but rather global functions that may be called through the use of the scope operator(`::`), e.g.

```
fileName::function_name();
```

The function declarations are in the respective header files while the function definitions reside in the accompanying implementation (`.cc`) files. It is the responsibility of the developer to write the function providing the logic to handle the parsed data. The existing Command “classes” provide many examples of how to write code to support a variety of different tokenized command strings.

## ▼ Adding Code to Support a New Command

With a minimum understanding of how the Parser class works, developers can rather easily write code to support a command interface for any object in the CUBIT environment. The necessary steps are briefly described below:

- write a global C++ function to “do the right thing” based on the tokenized data provided by the parser
- place the function prototype in a suitable header file, and the function definition in the implementation file
- pair up the conceptualized command string with the address of the function written above
- insert the command string/function pointer pair in the global CommandTable. (Hint: CUBIT’s CommandTable is growing rapidly, so

- (adding commands into the table alphabetically makes them a bit easier to maintain)
- inform the graphical user-interface developers so they can make plans to generate the command string through the Motif GUI.

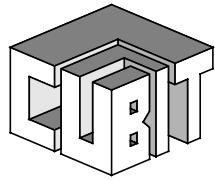
## ▼ Future Command Interface Development

CUBIT is currently moving toward a new command interface that will require changes to how new commands are supported. The majority of the old command grammar will remain the same; however, the logic to support the commands in code will change.

The Parser class is being replaced by the standard UNIX tools: Lex and YACC. The Lex and YACC specification being developed will directly call C++ class member functions in a manner similar to that used in the current Parser. Again, the parsed data will be passed as an argument of the function call. However, the member functions will now have to support multiple options on a single command line. Rather than a one-to-one matching of command strings and function pointers, the new command interface will have more of a “tree” structure.

The new command interface will also make better use of C++ and Object-Oriented Programming techniques. Polymorphism will be utilized so that the function pointers called will now be considered as generic “entity handlers”. The result will be a very reliable, flexible, yet easy to maintain program.





---

## Chapter 11 Miscellaneous Classes and Constants

▼ CubitVector

▼ CubitPlane

*The classes described in this chapter are at the top level of the class hierarchy and are not parent classes of any other class. They are typically support classes which perform specialized functions required by the other classes. The classes described below include CubitVector and CubitPlane which are typical three-dimensional vector and plane. Other classes which follow this chapter are AcisGeometryEngine which is an interface to ACIS used for geometry creation within CUBIT; CleanUp and ValidityCheck which fixup and validate the generated mesh; and ElementBlock\* which are related to managing the blocks of elements required by the Exodus II database.*

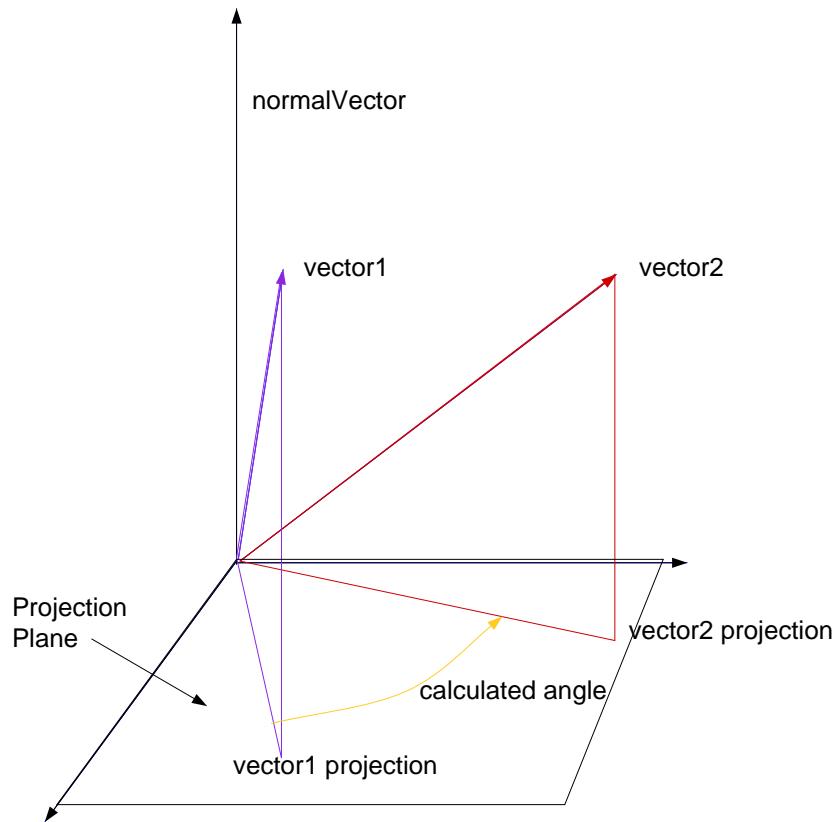
*Several files follow this chapter which define global constants for use throughout CUBIT.*

### ▼ CubitVector

A CubitVector is a standard three-dimensional vector consisting of an x, y, and z coordinate. They are used throughout CUBIT for direction and also interchangeably for locations in space. Because they are used as locations in space, CubitVectors are stored in an unnormalized format. All of the standard (addition, subtraction, multiplication, and division) arithmetic operators as well as a few vector-specific (cross product and dot product) operators are defined for the CubitVector class so they can be used similar to built-in types.

Two special functions (`vectorAngle` and `vectorRotate`) are defined for use in the Paving algorithm.

- `vectorAngle(normalVector, vector1, vector2)`  
computes the angle between the projections of `vector1` and `vector2` onto the plane defined by `normalVector`. The angle is computed in radians. A schematic of this is shown in Figure 3.



*Figure 3* Schematic of vectorAngle Function

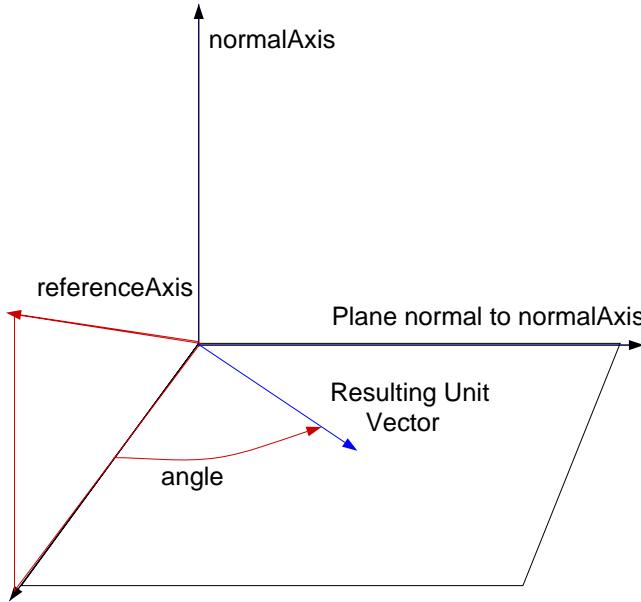
- `vectorRotate(angle, normalAxis, referenceAxis)`  
creates a new coordinate system with `normalAxis` corresponding to the Z axis and the X axis corresponding to the projection of `referenceAxis` onto the XY plane (normal to `normalAxis`). The returned vector is a unit vector angle radians from the X axis in the XY plane. A schematic of this is shown in Figure 4.

## ▼ CubitPlane

A **CubitPlane** defines a three-dimensional planar surface. It is currently only used in the **SweepTool** class. The equation of the surface is stored in the class. The equation is of the form:

$$Ax + By + Cz + D = 0$$

The normal to the plane is easily calculated as the vector  $\langle Ai + Bj + Ck \rangle$ . Currently, only the basic functionality required by **SweepTool** is implemented



*Figure 4* Schematic of vectorRotate Function

in the class; however, as the need arises new functions will be added. Two non-standard functions are provided:

- **CubitPlane(nodeList, use\_projection\_nodes)**

This constructor uses a technique devised by Martin Newell<sup>26</sup> to calculate the exact solution of the plane equation for planar polygons and the “best” approximation to the plane equation for non-planar polygons. The method is equivalent to determining the normal at each polygon vertex by taking the cross-product of the adjacent edges and averaging the results. In this case, the nodes in nodeList are assumed to define a polygon. The equations used in this method are:

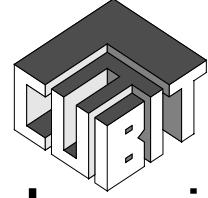
$$A = \sum_{i=1}^n \langle y_i - y_j \rangle \times \langle z_i + z_j \rangle$$

$$B = \sum_{i=1}^n \langle z_i - z_j \rangle \times \langle x_i + x_j \rangle$$

$$C = \sum_{i=1}^n \langle x_i - x_j \rangle \times \langle y_i + y_j \rangle$$

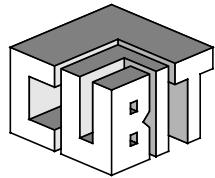
where  $j = i + 1$  if  $i < n$  otherwise  $j = 1$ . Variable D is calculated by averaging all of the nodal coordinates and then using the average value to solve for D such that the plane equation is satisfied. The variable `use_projection_nodes` is a boolean value that determines whether the coordinates of the actual nodes or the coordinates of the `projection_nodes` are used in the formulas.

- `report_plane_error(nodeList,use_projection_nodes)` calculates the average squared distance of the nodes in `nodeList` from the plane calculated as the best approximation to those nodes. This function gives an indication of the “goodness” of the plane to this set of nodes.



## Chapter 12 Overview of Graphics-Related Classes





---

# Class Owner/Author Index

CubitString 183  
DrawingTool 215  
StcEdge 557

## A

---

### Arlo Ames

FREDTool 287  
XpatchTool 765

## B

---

### Bill Bohnhoff

AttributeCommands 77  
CommandHandler 115  
CubitCollection 127  
CubitContainer 129  
CubitEntity 137  
CubitPtrArray 175  
CubitStack 181  
ElementBlock 253  
ElementBlock1D 257  
ElementBlock2D 259  
ElementBlock3D 261  
FileCommands 285  
GenesisCommands 295  
GenesisEntity 297  
GenesisTool 299  
MeshEntity 387  
Model 397  
ModelCommands 403  
NodeSet 415  
SideSet 541  
SLLList 547  
TriangleTool 671

### Brett Clark

Mouse 407

## D

---

### Dan Goodrich

CubitMatrix 153  
Pyramid 443  
PyramidTool 445  
SurfMorphTool 585  
TDLaplace 611  
TDMorph 619

### David White

AutoVertexType 79  
MapToolSupport 377  
SubMapNode 565  
SubMapTool 569  
SurfVertexType 593  
TDNodeHardLine 621  
TDSubMap 637  
VolMapTool 687  
VolSubMapTool 693  
VolumeEdgeType 699

## G

---

### Greg Sjaardema

CubitBox 125  
CubitPlane 171  
CubitString 183  
CubitVector 185  
database.hpp 197  
DLLList 199  
FastqCommands 283  
FastqEntity 277  
GetLongOpt 319  
MeshTool 395  
ModelViewer 405  
QualityCommands 451  
RefEntityName 479  
RefEntityNameMap 481  
RotateTool 509  
SDLLList 511  
SweepTool 595  
TDCenterNode 599

# Class Owner/Author Index

TDGenesisID 605  
TDLayer 613  
TDWeight 651  
TranslateTool 665  
TwoHalfDimTool 675  
VolMeshTool 691

## J

---

### Jan Clements

BoundaryLayer 93  
BoundaryLayerEdge 95  
BoundaryLayerFace 99  
BoundaryLayerTool 101

### Jim Hipp

GridSearch 323  
HexTool 345  
MemoryBlock 381  
MemoryManager 383  
MeshIntersect 391  
ProtoHex 437  
UserInterface 681

## M

---

### Mark Whiteley

SurfMapTool 575

### Mark Whiteley, Scott Mitchell modified 6-20-96

MappingFace 373

## P

---

### Paul Kinney

ArrayBasedContainer 63  
DynamicArray 245

### Phil Tuchinsky

BaseHexer 81  
HexToVoid 347  
VolVoidBoundary 701  
VolVoidSepTool 703

## R

---

### Randy Lober

CubitKnife 147  
CubitNodeArray 170  
CubitNodeArray1D 167  
CubitNodeArray2D 169  
EdgeMeshTool 249  
EdgeUse 251

FaceUse 273  
Hexer 329  
NodeHex 411  
ParallelMeshTool 419  
RefEdge 465

### Ray Ostensen

TDCellIndex 597  
TDDistance 603

### Rob Oakes

AcisGeometryEngine 55  
GeometryTool 309

## S

---

### S Manoharan, Advanced Computer Research Institute, Lyon, France

GetLongOpt 319

### Scott Mitchell

BladeEdges 89  
ControlPoint 119  
DoubletPillower 205  
DoubletPillowerTD 211  
HexKnowingNode 339  
IntervalLinearProgram 351  
LinearProgram 357  
LoopCollapser 365  
LoopInterval 369  
LoopJoiner 371  
MuseOutput 409  
PillowNode 431  
PillowSheet 433  
Queue 457  
SelfCrossing 515  
SelfCrossingLoop 517  
SheetEdge 535  
StarNode 555  
TDCopied 601  
TDGeometrySizing 607  
TDHexKnowing 609  
TDLPEdge 615  
TDNormal 623  
TDPaver 625  
TDPillow 627  
TDProjection 631  
TDSizing 633  
TDStrider 635  
TDTipton 643

# Class Owner/Author Index

TDUVSpace 647  
TDVolMapTool 649  
ToolData 659  
ToolDataUser 663  
Tree 667  
WhiskerKnife 735

## Steve Owen

CubitVoxel 191  
SurfDistributeTool 573  
VoxelTool 705

## T

---

## Ted Blacker

CubitFace 139  
CubitHex 143  
CubitLogical.hpp 151  
CubitSheet 179  
DynLoopList 247  
Loop 363  
MeshingCommands 389  
RefBody 461  
RefEntity 471  
RefGroup 495  
RefVertex 497  
RefVolume 501  
SurfMeshTool 579  
Tool 657  
TriElementTool 673  
WhiskerChord 713  
WhiskerFace 723  
WhiskerSheet 739  
WhiskerWeaver 753

## Tim Tautges

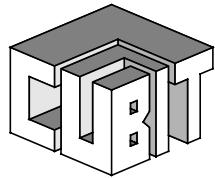
CubitDefines 131  
CubitMessage 155  
CubitNode 161  
ElementQuality 263  
ExodusMesh 265  
FullHex 289  
GeometryCommands 303  
HexQuality 341  
QuadQuality 447  
SheetChord 519  
SheetChordPoint 525  
StairTool 551  
WhiskerCell 709

WhiskerHex 727  
**Tony Edwards**  
GraphicsCommands 321  
**Tony L. Edwards**  
DrawingToolInstance 227

# Class Owner/Author Index

# Class Owner/Author Index





---

# Class Index

## A

---

**AbsTime**

*Base Class*  
    TimeVal  
*Class Declaration* 654  
*Used In*  
    DeltaTime 655  
    TimeVal 653

**AcisGeometryEngine**

*Class Declaration* 55  
*Used In*  
    FastqCommands 284  
    GeometryTool 311, 315  
    RefFace 486, 493

**ArrayBasedContainer**

*Class Declaration* 63  
*Derived Classes*  
    DLLList 199  
    DynamicArray 245  
*Used In*  
    DLLList 202, 204  
    DynamicArray 245, 246  
    SDLLList 512, 513

**ArrayMemory**

*Class Declaration* 64

**ATTRIB\_CUBIT\_OWNER**

*Base Class*  
    ATTRIB\_SNL  
*Class Declaration* 67

**ATTRIB\_GTC**

*Derived Classes*  
    ATTRIB\_GTC\_NAME 71

**ATTRIB\_GTC\_NAME**

*Base Class*  
    ATTRIB\_GTC  
*Class Declaration* 71  
*Used In*

AcisGeometryEngine 59, 61

**ATTRIB\_PARENTS**

*Base Class*  
    ATTRIB\_SNL  
*Class Declaration* 73

**ATTRIB\_SNL**

*Derived Classes*  
    ATTRIB\_CUBIT\_OWNER 67  
    ATTRIB\_PARENTS 73

**AttributeCommands**

*Base Class*  
    CommandHandler  
*Class Declaration* 77

**AutoVertexType**

*Class Declaration* 79

---

## B

---

**BaseHexer**

*Base Class*  
    HexTool  
*Class Declaration* 81  
*Derived Classes*  
    Hexer 329  
    HexToVoid 347  
*Used In*  
    Hexer 329, 330, 337  
    HexToVoid 347, 348, 349, 350  
    ProtoHex 437, 438, 439, 441  
    SortedNodeDLLList\* 86, 87

**BladeEdges**

*Class Declaration* 89  
*Used In*  
    BladeEdgesTree.hpp 91  
    WhiskerKnife 735, 736, 737

**BladeEdgesTree**

*Used In*  
    BladeEdgesTree.hpp 91  
    WhiskerKnife 735, 736, 737

# Class Index

## BODY

### *Used In*

AcisGeometryEngine 61  
GeometryCommands 304  
GeometryTool 315  
Model 400  
RefBody 463

## BoundaryLayer

### *Base Class*

MeshEntity

### *Class Declaration* 93

### *Used In*

BoundaryLayerEdge 95, 96  
BoundaryLayerFace 99, 100  
BoundaryLayerTool 101, 102  
Model 397, 398, 400

## BoundaryLayerEdge

### *Base Class*

CubitEntity

### *Class Declaration* 95

### *Used In*

BoundaryLayerFace 99, 100  
BoundaryLayerTool 101, 102  
EdgeMeshTool 249, 250  
RefEntity 474, 476

## BoundaryLayerFace

### *Class Declaration* 99

### *Used In*

Model 397, 398, 400

## BoundaryLayerTool

### *Base Class*

SurfMeshTool

### *Class Declaration* 101

## C

---

## CleanUp

### *Base Class*

SurfMeshTool

### *Class Declaration* 105

### *Used In*

SurfMeshTool 582

## COEDGE

### *Used In*

AcisGeometryEngine 61

## CommandHandler

### *Class Declaration* 115

## *Derived Classes*

AttributeCommands 77  
FastqCommands 283  
FileCommands 285  
GenesisCommands 295  
GeometryCommands 303  
GraphicsCommands 321  
MeshingCommands 389  
ModelCommands 403  
QualityCommands 451

### *Used In*

UserInterface 682, 683

## ControlPoint

### *Class Declaration* 119

## CopiedData

### *Class Declaration* 121

## CpuTimer

### *Class Declaration* 123

## CubitBox

### *Class Declaration* 125

### *Used In*

AcisGeometryEngine 59, 61  
RefEntity 473, 476  
RefGroup 496

## CubitCollection

### *Base Class*

CubitContainer

### *Class Declaration* 127

### *Derived Classes*

CubitPtrArray1D 175  
CubitPtrArray2D 176  
CubitPtrArray3D 177  
SLLList 548

## CubitContainer

### *Class Declaration* 129

### *Derived Classes*

CubitCollection 127  
CubitStack 181

## CubitEdge

### *Base Class*

MeshEntity

### *Class Declaration* 133

### *Used In*

CleanUp 106, 108, 111, 112, 113  
CubitFace 139, 140, 142  
CubitHex 144

# Class Index

- CubitKnife 148, 149
- CubitNode 163, 164, 166
- CubitVoxel 191, 192
- DoubletPillower 207, 208
- DrawingTool 222, 223, 224, 225
- DrawingToolInstance 234, 239, 240, 241, 242
- EdgeUse 251, 252
- ExodusMesh 268, 269
- FaceUse 273, 274, 275
- GenesisTool 300, 301
- GeometryTool 315
- GridSearch 323, 325, 327
- Hexer 330, 333, 334, 335, 336, 337
- HexToVoid 347, 348, 350
- MeshIntersect 391, 392, 393
- Paver 426, 427, 428, 429
- PillowSheet 434, 435
- ProtoHex 439, 441
- Pyramid 443
- PyramidTool 445
- RefEdge 467, 469
- RefEntity 473, 476
- RefFace 484, 485, 487, 492, 493
- RefVolume 504, 506
- SortedNodeDLLList\* 83, 84, 86, 87
- SurfMeshTool 579, 581, 582
- SurfMorphTool 585, 586
- TDLayer 613, 614
- TDNodeHardLine 621, 622
- TDPaver 625, 626
- VolMapTool 687, 688
- VolSubMapTool 694, 697
- VoxelTool 706
- WhiskerCell 709, 711, 712
- WhiskerFace 724, 725
- WhiskerKnife 736, 737
- WhiskerSheet 740, 748, 752
- CubitEntity**
  - Class Declaration* 137
  - Derived Classes*
    - BoundaryLayerEdge 95
    - FaceUse 273
    - GenesisEntity 297
    - MeshEntity 387
    - RefEntity 471
- Used In**
  - function\_templates.hpp 293
  - GenesisEntity 298
  - GenesisTool 301
  - RefEntity 477
  - RefEntityName 480
- CubitFace**
  - Base Class*
    - MeshEntity
  - Class Declaration* 139
  - Used In*
    - CleanUp 106, 108, 109, 111, 112, 113
    - CubitEdge 133, 134
    - CubitHex 144
    - CubitKnife 148, 149
    - CubitNode 164, 165, 166
    - CubitSheet 179
    - DoubletPillower 206, 207, 208
    - DrawingTool 222, 223, 224, 225
    - DrawingToolInstance 234, 240, 241, 242
    - EdgeUse 251, 252
    - ExodusMesh 266, 268, 269
    - FaceUse 273, 274, 275
    - FullHex 290, 291
    - GenesisTool 300, 301
    - GeometryTool 315
    - GridSearch 323, 325, 327
    - Hexer 330, 333, 334, 335, 336, 337
    - HexToVoid 349, 350
    - MeshEntity 387
    - MeshIntersect 391, 392, 393
    - NodeHex 412, 413
    - Paver 426, 427, 428, 429
    - PillowSheet 434, 435
    - ProtoHex 437, 438, 439, 441
    - Pyramid 443
    - PyramidTool 445
    - RefEntity 473, 476
    - RefFace 487, 493
    - RefVolume 504, 506
    - SelfCrossing 515
    - SelfCrossingLoop 517, 518
    - SortedNodeDLLList\* 83, 84, 86, 87
    - StairTool 551, 553
    - ToolDataUser 663

# Class Index

TwoHalfDimTool 676, 679  
VolMapTool 688  
VolSubMapTool 694, 695, 697  
VolVoidBoundary 701  
VolVoidSepTool 703, 704  
WhiskerFace 723, 724, 725  
WhiskerKnife 736, 737  
WhiskerSheet 740, 741, 742, 743, 745,  
    747, 751, 752  
WhiskerWeaver 760, 762

## CubitHex

*Base Class*  
    MeshEntity  
*Class Declaration* 143  
*Derived Classes*  
    FullHex 289  
    NodeHex 411  
*Used In*  
    CubitFace 140, 142  
    CubitKnife 148, 149  
    DoubletPillower 206, 208  
    DrawingTool 222, 225  
    DrawingToolInstance 231, 232, 240,  
        242  
    FaceUse 273, 274, 275  
    GridSearch 323, 325, 327  
    Hexer 330, 333, 337  
    HexTool 346  
    HexToVoid 348, 350  
    Mouse 407, 408  
    RefEntity 472, 473, 476  
    RefVolume 503, 504, 505, 506  
    SortedNodeDLLList\* 84, 87  
    WhiskerHex 727, 733

## CubitKnife

*Base Class*  
    MeshEntity  
*Class Declaration* 147  
*Used In*  
    FaceUse 273, 274, 275  
    Hexer 332, 337  
    NodeHex 412, 413

## CubitMatrix

*Class Declaration* 153  
*Used In*  
    SurfMorphTool 585, 586

## CubitMessage

*Class Declaration* 155

## CubitNode

*Base Class*

    MeshEntity

*Class Declaration* 161

*Used In*

    BaseHexer 82

    BoundaryLayerTool 102

    CleanUp 105, 106, 107, 108, 109, 110,  
        111, 112, 113

    CubitBox 125, 126

    CubitEdge 133, 134

    CubitFace 139, 140, 141, 142

    CubitHex 143, 144, 145

    CubitKnife 147, 148, 149

    CubitNodeArray1D 167, 168

    CubitPlane 172

    CubitVector 186

    CubitVoxel 191, 192, 193

    DoubletPillower 206, 207, 208

    DoubletPillowerTD 211, 212, 213

    DrawingTool 222, 223, 224, 225

    DrawingToolInstance 230, 231, 234,  
        239, 240, 241, 242

    EdgeMeshTool 249, 250

    EdgeUse 251, 252

    ExodusMesh 266, 267, 268, 269

    FaceUse 273, 274, 275

    FullHex 289, 290, 291

    GenesisTool 300, 301

    GeometryTool 313, 314, 315

    GridSearch 323, 324, 325, 326, 327

    Hexer 330, 332, 333, 334, 335, 336,  
        337

    HexKnowingNode 339

    HexTool 345, 346

    HexToVoid 347, 348, 349, 350

    MapToolSupport 377, 378

    MeshTool 395

    NodeHex 411, 412, 413

    ParallelMeshTool 419, 420

    Paver 423, 425, 427, 428, 429

    Pillownode 431, 432

    ProtoHex 439, 441

    Pyramid 443

# Class Index

- PyramidTool 445
- RefEdge 467, 468, 469
- RefEntity 473, 476
- RefFace 484, 485, 487, 488, 489, 491, 492, 493
- RefVertex 497, 498, 499
- RefVolume 504, 506
- SortedNodeDLLList\* 83, 84, 85, 86, 87
- StairTool 551, 553
- Strider 561
- SubMapNode 565, 566
- SurfMapTool 577, 578
- SurfMeshTool 579, 580, 581, 582
- SurfMorphTool 585, 586
- TDCenterNode 599
- TDCopied 601
- TDLaplace 611, 612
- TDMorph 619, 620
- TDNodeHardLine 621, 622
- TDNormal 623, 624
- TDPillow 627, 628
- TDPProjection 631
- TDSubMap 637, 638, 641
- ToolData 659, 660
- ToolDataUser 663
- TriangleTool 671
- TwoHalfDimTool 676, 678, 679
- ValidityCheck.hpp 685
- VolMapTool 687, 688
- VoxelTool 706
- WhiskerCell 709, 712
- WhiskerFace 725
- WhiskerHex 730, 731, 733, 734
- WhiskerSheet 743, 752
- WhiskerWeaver 759, 760, 762
- CubitNodeArray**
  - Class Declaration* 170
  - Used In*
    - CubitNodeArray1D 167, 168
    - CubitPtrArray 175
    - CubitPtrArray1D 175
    - MapToolSupport 377, 378
    - SurfMapTool 575, 578
    - VolMapTool 687, 688
- CubitNodeArray1D**
  - Class Declaration* 167
- CubitNodeArray2D**
  - Class Declaration* 169
  - Used In*
    - CubitNodeArray 170
- CubitPlane**
  - Class Declaration* 171
  - Used In*
    - SweepTool 595
    - TwoHalfDimTool 678, 679
- CubitPtrArray1D**
  - Base Class*
    - CubitCollection
  - Class Declaration* 175
  - Used In*
    - CubitPtrArray2D 176
- CubitPtrArray2D**
  - Base Class*
    - CubitCollection
  - Class Declaration* 176
  - Used In*
    - CubitPtrArray3D 177
- CubitPtrArray3D**
  - Base Class*
    - CubitCollection
  - Class Declaration* 177
- CubitSheet**
  - Base Class*
    - MeshEntity
  - Class Declaration* 179
  - Used In*
    - Model 398, 400
- CubitStack**
  - Base Class*
    - CubitContainer
  - Class Declaration* 181
- CubitString**
  - Class Declaration* 183
  - Used In*

# Class Index

- AcisGeometryEngine 55, 59, 60, 61
- CommandHandler 115, 116
- CubitMessage 155, 156, 157
- DrawingTool 215, 225
- DrawingToolInstance 234, 242
- GenesisEntity 298
- GenesisTool 299, 301
- MessageFlag 155
- RefEntity 472, 476
- RefEntityName 479, 480
- RefEntityNameMap 481
- RotateTool 509, 510
- SweepTool 595
- TranslateTool 665, 666
- TwoHalfDimTool 675, 679
- UserInterface 682, 683
- CubitStringRep**
  - Used In*
    - CubitString 184
- CubitVector**
  - Used In*
    - AcisGeometryEngine 57, 60, 61
    - BoundaryLayerEdge 96
    - BoundaryLayerTool 102
    - CleanUp 106, 112, 113
    - CubitBox 125, 126
    - CubitFace 140, 142
    - CubitHex 144, 145
    - CubitKnife 147, 149
    - CubitMatrix 153, 154
    - CubitNode 162, 163, 164, 166
    - CubitPlane 171, 172
    - CubitVoxel 192, 193
    - DoubletPillower 207, 208
    - DrawingTool 219, 220, 222, 225
    - DrawingToolInstance 233, 236, 237, 238, 239, 240, 242
    - ExodusMesh 265, 269
    - FaceUse 274, 275
    - FastqCommands 283, 284
    - GeometrySizingTool 307
    - GeometryTool 310, 311, 314, 316
    - GraphicsCommands 321, 322
    - GridSearch 323, 324, 325, 326, 327
    - Hexer 330, 332, 334, 337
    - HexToVoid 347, 349, 350
- MeshIntersect 391, 392, 393
- ParallelMeshTool 420
- Paver 425, 427, 429
- ProtoHex 439, 441
- PyramidTool 445
- QuadQuality 447, 448
- RefBody 462, 463
- RefEdge 465, 467, 468, 469
- RefEntity 473, 476
- RefFace 483, 484, 485, 487, 488, 492, 493
- RefVertex 498, 499
- RefVolume 504, 506
- RotateTool 509, 510
- SheetChord 520, 524
- SheetChordPoint 525, 526, 528, 529, 533
- SortedNodeDLLList\* 83, 86, 87
- StairTool 552, 553
- Strider 561
- SurfMeshTool 579, 580, 581, 582
- TDNormal 623, 624
- TDTipton 643, 644
- TranslateTool 666
- TriangleTool 671
- TwoHalfDimTool 675, 678, 679
- WhiskerChord 716, 720
- WhiskerHex 727, 730, 732, 734
- WhiskerSheet 740, 743, 745, 750, 752
- CubitVoxel**
  - Class Declaration* 191
  - Used In*
    - VoxelTool 705, 706
- curve**
  - Used In*
    - RefEdge 470
- CurveSubDomain**
  - Base Class*
    - SubDomain
  - Class Declaration* 195
  - Used In*
    - SurfSubDomain 588
- D**

---
- DeltaTime**
  - Base Class*

# Class Index

- TimeVal
  - Class Declaration* 655
  - Used In*
    - AbsTime 654
    - TimeVal 653
- DLBlaDeEdgesList**
  - Used In*
    - WhiskerKnife 737
- DLBLEDgeLoopList**
  - Used In*
    - BoundaryLayerFace 99, 100
- DLBoundaryLayerEdgeList**
  - Used In*
    - BoundaryLayerFace 99, 100
    - BoundaryLayerTool 101, 102
- DLBoundaryLayerFaceList**
  - Used In*
    - Model 397, 399, 400
- DLBoundaryLayerList**
  - Used In*
    - BoundaryLayerTool 101, 102
    - Model 397, 399, 400
- DLChordPointList**
  - Used In*
    - PillowSheet 433, 435
    - SheetChord 519, 521, 522, 524
    - SheetChordPoint 528, 533
    - WhiskerCell 709, 711, 712
    - WhiskerChord 713, 714, 720
    - WhiskerSheet 739, 741, 743, 744, 747, 748, 752
    - WhiskerWeaver 753, 756, 762
- DLCopiedDataList**
  - Used In*
    - RefEntity 471, 475, 476
- DLCubitEdgeList**
  - Used In*
    - BaseHexer 81
    - CleanUp 106, 107, 108, 109, 110, 113
    - CubitFace 140, 142
    - CubitHex 143, 145
    - CubitNode 163, 166
    - ExodusMesh 265, 268, 269
    - FullHex 290, 291
    - GridSearch 325, 327
    - Hexer 332, 337
- MeshIntersect** 392, 393
- Model** 399, 400
- Paver** 426, 427, 428, 429
- RefEdge** 465, 466, 469
- RefFace** 484, 486, 493
- RefVolume** 501, 503, 506
- SideSet** 542
- SortedNodeDLLList\*** 87
- TDLaplace** 611, 612
- TDPaver** 625, 626
- TwoHalfDimTool** 677, 679
- VolSubMapTool** 694, 695, 697
- DLCubitEntityList**
  - Used In*
    - function\_templates.hpp 293
    - GraphicsCommands 321, 322
    - Model 399, 400
- DLCubitFaceList**
  - Derived Classes*
    - VolVoidBoundary 701
  - Used In*
    - BaseHexer 81
    - CleanUp 106, 107, 108, 109, 110, 113
    - CubitEdge 133, 134
    - CubitFace 142
    - CubitKnife 147, 148, 149
    - CubitNode 163, 164, 165, 166
    - CubitSheet 179
    - ExodusMesh 265, 268, 269
    - GenesisTool 301
    - GridSearch 325, 328
    - Hexer 329, 332, 333, 337
    - HexToVoid 347, 350
    - LoopCollapser 366
    - MeshIntersect 392, 393
    - Model 399, 400, 401
    - ParallelMeshTool 419, 420
    - ProtoHex 439, 441
    - PyramidTool 445
    - RefFace 484, 486, 493
    - RefVolume 501, 503, 504, 506
    - RotateTool 510
    - SelfCrossingLoop 518
    - SideSet 542
    - SortedNodeDLLList\* 82, 83, 86, 87
    - StairTool 551, 552, 553

# Class Index

SubMapTool 572  
SurfMeshTool 579, 581, 582  
SurfMorphTool 585, 586  
SurfSubDomain 587, 588  
SweepTool 595  
TranslateTool 666  
TwoHalfDimTool 676, 677, 679  
VolMapTool 688  
VolSubMapTool 695, 697  
VolVoidBoundary 701  
VolVoidSepTool 703, 704  
WhiskerFace 724, 725  
WhiskerSheet 739, 742, 748, 752  
WhiskerWeaver 756, 762

## DLCubitHexList

### *Used In*

CubitEdge 133, 134  
CubitFace 141, 142  
CubitKnife 147, 149  
CubitNode 164, 166  
DoubletPillower 206, 207, 208  
GridSearch 325, 328  
Hexer 329, 337  
HexKnowingNode 339  
HexTool 345, 346  
HexToVoid 347, 350  
Model 399, 400, 401  
Mouse 407, 408  
RefVolume 501, 503, 506  
SortedNodeDLLList\* 82, 87  
SurfMeshTool 579, 582  
TDHexKnowing 609

## DLCubitNodeList

### *Derived Classes*

Loop 363

### *Used In*

AcisGeometryEngine 58, 61  
BaseHexer 81  
BoundaryLayerEdge 95, 96  
BoundaryLayerTool 101, 102  
CleanUp 105, 106, 107, 108, 109, 110, 111, 113  
CubitFace 140, 142  
CubitKnife 148, 149  
CubitNode 163, 166  
CubitPlane 171, 172

CubitVoxel 192, 193  
DoubletPillower 206, 207, 208  
ExodusMesh 265, 266, 268, 269  
FaceUse 274, 275  
FullHex 289, 291  
GeometryTool 315, 316  
GridSearch 323, 324, 325, 327, 328  
Hexer 330, 332, 333, 335, 337  
HexTool 345, 346  
HexToVoid 348, 350  
MapToolSupport 377, 378  
Model 399, 400, 401  
NodeHex 412, 413  
NodeSet 415, 417  
ParallelMeshTool 419, 420  
Paver 425, 426, 427, 429  
Pillownode 431, 432  
RefEdge 465, 466, 469  
RefFace 484, 486, 487, 493  
RefVolume 501, 503, 506  
RotateTool 510  
SortedNodeDLLList\* 84, 86, 87  
StarNode 555  
Strider 561  
SubMapTool 570, 572  
SurfMapTool 575, 577, 578  
SurfMeshTool 579, 580, 581, 582  
SurfMorphTool 585, 586  
SweepTool 595  
TDLaplace 611, 612  
TDNodeHardLine 622  
TDPillow 627, 628  
TranslateTool 665, 666  
TriangleTool 671, 672  
TwoHalfDimTool 676, 677, 678, 679  
VolMapTool 687, 688  
WhiskerHex 727, 730, 732, 734

## DLCubitSheetList

### *Used In*

Model 397, 399, 400, 401

## DLCubitStringList

### *Used In*

RefEntity 472, 476  
RefEntityName 479, 480

## DLCubitVectorList

### *Used In*

# Class Index

- CubitPlane 171, 172
- WhiskerSheet 739, 741, 750, 752
- DLCurveList**
  - Used In*
    - RefEdge 468, 469
    - RefFace 488, 493
- DLCurveSubDomainList**
  - Used In*
    - SurfSubDomain 587, 588
- DLDoubleList**
  - Used In*
    - BoundaryLayer 93, 94
    - BoundaryLayerEdge 95, 96
    - MappingFace 375
- DLEDGEList**
  - Used In*
    - GeometryTool 315, 316
    - RefEdge 465, 468, 469
- DLEdgeUseList**
  - Used In*
    - CubitEdge 133, 134
- DLElementQualityList**
  - Used In*
    - QualityController 453
- DLFACEList**
  - Used In*
    - AcisGeometryEngine 56, 61
    - GeometryTool 315, 316
    - RefFace 483, 488, 493
- DLFaceUseList**
  - Used In*
    - CubitEdge 133, 134
    - CubitFace 139, 140, 142
    - CubitKnife 147, 149
    - CubitNode 166
    - Hexer 329, 337
    - HexToVoid 347, 350
    - SortedNodeDLLList\* 82, 87
- DLFastqBCList**
  - Used In*
    - FastqCommands 283, 284
    - FastqModel 277
    - FastqSide 280
- DLFastqEntityList**
  - Used In*
    - FastqBC 279
- FastqCommands 283, 284
- FastqRegion 279
- FastqSide 280
- DLFastqLineList**
  - Used In*
    - FastqCommands 283, 284
    - FastqModel 277
    - FastqRegion 279
    - FastqSide 280
- DLFastqPointList**
  - Used In*
    - FastqCommands 283, 284
    - FastqModel 277
    - FastqSide 280
- DLFastqRegionList**
  - Used In*
    - FastqCommands 283, 284
    - FastqModel 277
    - FastqRegion 279
    - FastqSide 280
- DLFastqSideList**
  - Used In*
    - FastqCommands 283, 284
    - FastqModel 277
    - FastqSide 280
- DLFilePtrList**
  - Used In*
    - UserInterface 682, 683
- DLGensisEntityList**
  - Used In*
    - function\_templates.hpp 293
    - GenesisCommands 295
    - GenesisTool 299, 300, 301
    - Model 398, 401
- DLIntList**
  - Used In*
    - LoopCollapser 365, 366
    - SurfMapTool 576, 578
- DLList**
  - Base Class*
    - ArrayBasedContainer
  - Class Declaration* 199
  - Derived Classes*
    - SDLList 511
  - Used In*
    - DynamicArray 245, 246

# Class Index

LoopCollapser 365, 366

LoopJoiner 371

RefEntityName 480

Tree 667, 669

WhiskerWeaver 755, 760, 762

## DLLoopList

*Used In*

CleanUp 105, 113

Paver 423, 426, 427, 429

## DLMeshEntityList

*Used In*

function\_templates.hpp 293

Model 398, 401

RefEdge 466, 469

RefEntity 473, 476

RefFace 486, 493

RefGroup 495, 496

RefVertex 498, 499

RefVolume 503, 506

WhiskerWeaver 758, 762

## DLMeshIntersectList

*Used In*

Hexer 332, 337

HexToVoid 349, 350

SortedNodeDLLList\* 86, 87

## DLNodeLoopList

*Used In*

BoundaryLayerTool 101, 102

CleanUp 105, 113

CubitVoxel 192, 193

MapToolSupport 378

Paver 423, 428, 429

RefFace 484, 486, 493

SurfDistributeTool 573

SurfMapTool 577, 578

SurfMeshTool 580, 581, 582

SurfMorphTool 585, 586

TriElementTool 673

VoxelTool 706

## DLPillowSheetList

*Used In*

PillowSheet 433, 434, 435

## DLProtoHexList

*Used In*

BaseHexer 81

Hexer 329, 330, 337

HexToVoid 347, 350

SortedNodeDLLList\* 82, 85, 87

## DLRefBodyList

*Used In*

GeometryCommands 303, 304

GeometryTool 309, 310, 312, 313, 316

Model 397, 399, 400, 401

## DLRefEdgeList

*Used In*

AcisGeometryEngine 57, 58, 61

AutoVertexType 79

BoundaryLayerFace 99, 100

ExodusMesh 265, 266, 268, 269

FastqCommands 283, 284

FastqLine 278

FastqSide 280

GeometryTool 312, 313, 314, 316

MappingFace 373, 374, 375

Model 397, 399, 400, 401

ModelViewer 405, 406

NodeSet 415, 416, 417

RefBody 462, 463

RefFace 486, 493

RefGroup 496

RefVertex 499

RefVolume 503, 506

SideSet 542

SizingTool 545

SubMapTool 572

SurfMapTool 575, 578

SurfMorphTool 585, 586

SurfSubDomain 587, 588

TriElementTool 673

TwoHalfDimTool 678, 679

VolSubMapTool 693, 694, 697

## DLRefEntityList

*Used In*

AcisGeometryEngine 60, 61

AttributeCommands 77

function\_templates.hpp 293

GeometrySizingTool 307

GeometryTool 314, 316

HexTool 346

Model 398, 401

ModelCommands 403

NodeSet 415, 417

# Class Index

QualityCommands 451  
RefBody 462, 463  
RefEdge 469  
RefEntity 474, 476  
RefFace 490, 493  
RefGroup 495, 496  
RefVertex 499  
RefVolume 505, 506  
SideSet 542  
SurfMeshTool 581, 582

## DLRefFaceList

*Used In*  
AttributeCommands 77  
BoundaryLayerTool 101, 102  
ExodusMesh 265, 269  
GeometryCommands 304  
GeometryTool 310, 312, 313, 315, 316  
HexToVoid 347, 350  
Model 397, 399, 400, 401  
NodeSet 415, 416, 417  
RefBody 462, 463  
RefEdge 468, 469  
RefEntity 472, 476  
RefGroup 496  
RefVolume 501, 502, 503, 505, 506  
RotateTool 509, 510  
SideSet 542  
SurfMapTool 575, 577, 578  
SurfMorphTool 585, 586  
SweepTool 595  
TranslateTool 665, 666  
TwoHalfDimTool 675, 676, 679

## DLRefGroupList

*Used In*  
Model 397, 399, 400, 401  
RefEntity 471, 476

## DLRefVertexList

*Used In*  
AutoVertexType 79  
ExodusMesh 265, 269  
FastqCommands 283, 284  
FastqPoint 278  
FastqSide 280  
GeometryTool 312, 313, 316  
Model 397, 399, 400, 401  
NodeSet 415, 416, 417

RefBody 462, 463  
RefEdge 469  
RefFace 483, 486, 489, 493  
RefGroup 496  
RefVolume 504, 506  
TwoHalfDimTool 678, 679  
VolMapTool 688

## DLRefVertLoopList

*Used In*  
RefFace 486, 493

## DLRefVolumeList

*Used In*  
FREDTool 287, 288  
GenesisTool 299, 300, 301  
Model 397, 399, 400, 401  
NodeSet 415, 416, 417  
RefBody 462, 463  
RefFace 488, 493  
RefGroup 496  
XpatchTool 765, 766

## DLSelfCrossingList

*Derived Classes*  
SelfCrossingLoop 517

*Used In*  
LoopCollapser 365, 366  
LoopJoiner 371  
SelfCrossing 515  
SelfCrossingLoop 517, 518

## DLSheetChordList

*Used In*  
SheetChord 520, 521, 524  
WhiskerSheet 739, 740, 741, 742, 748, 749, 752  
WhiskerWeaver 753, 758, 762

## DLSheetEdgeList

*Used In*  
WhiskerKnife 736, 737

## DLStcEdgeList

*Used In*  
StcEdge 557, 558, 559  
WhiskerCell 711, 712

## DLSubLoopList

*Used In*  
MapToolSupport 378  
SubMapTool 569, 570, 572

## DLSurfSubDomainList

# Class Index

- Used In*
  - ParallelMeshTool 419, 420
- DLVERTEXList**
  - Used In*
    - RefVertex 497, 498, 499
- DLVOLEdgeList**
  - Used In*
    - RefVolume 506
- DLVOLVoidBoundaryList**
  - Used In*
    - VolVoidSepTool 703, 704
- DLWhiskerCellList**
  - Used In*
    - StcEdge 557, 558, 559
    - WhiskerCell 711, 712
    - WhiskerFace 723, 724, 725
    - WhiskerHex 727, 730, 732, 734
    - WhiskerSheet 747, 752
    - WhiskerWeaver 753, 755, 759, 760, 762
- DLWhiskerChordList**
  - Used In*
    - PillowSheet 433, 435
    - WhiskerChord 716, 720
    - WhiskerHex 729, 734
    - WhiskerSheet 740, 747, 752
    - WhiskerWeaver 753, 754, 755, 757, 758, 760, 762
- DLWhiskerFaceList**
  - Used In*
    - PillowSheet 433, 434, 435
- DLWhiskerHexList**
  - Used In*
    - Model 399, 400, 401
    - PillowSheet 433, 434, 435
    - WhiskerCell 710, 712
    - WhiskerChord 713, 714, 715, 718, 720
    - WhiskerSheet 744, 752
    - WhiskerWeaver 759, 760, 762
- DLWhiskerSheetList**
  - Used In*
    - Model 399, 400, 401
    - WhiskerWeaver 753, 755, 756, 759, 760, 762
- DoubletPillower**
  - Base Class*
    - DoubletPillowerTD
    - HexTool
  - Class Declaration* 206
  - Used In*
    - DoubletPillowerTD 211, 213
- DoubletPillowerTD**
  - Class Declaration* 211
  - Derived Classes*
    - DoubletPillower 206
- DrawingTool**
  - Base Class*
    - Tool
  - Class Declaration* 215
  - Used In*
    - CubitDefines 131
    - DrawingToolInstance 227, 242
    - GraphicsCommands 321, 322
    - VolVoidBoundary 701
- DrawingToolInstance**
  - Base Class*
    - Tool
  - Class Declaration* 227
  - Used In*
    - DrawingTool 215, 225
- DynamicArray**
  - Base Class*
    - ArrayBasedContainer
  - Class Declaration* 245
- E**

---
- EDGE**
  - Used In*
    - AcisGeometryEngine 61
    - BoundaryLayerEdge 96
    - CubitFace 142
    - Model 401
    - RefFace 493
- EdgeMeshTool**
  - Base Class*
    - MeshTool
  - Class Declaration* 249
  - Derived Classes*
    - GeometrySizingTool 307
    - Strider 561
  - Used In*
    - GeometrySizingTool 307

# Class Index

- SizingTool 545
- EdgeUse**
  - Class Declaration* 251
  - Used In*
    - CubitEdge 133, 134
    - CubitFace 139, 141, 142
    - CubitNode 166
    - FaceUse 274, 275
    - Hexer 337
- ElementBlock**
  - Base Class*
    - GenesisEntity
  - Class Declaration* 253
  - Derived Classes*
    - ElementBlock1D 257
    - ElementBlock2D 259
    - ElementBlock3D 261
  - Used In*
    - CubitHex 143, 145
    - DrawingTool 225
    - DrawingToolInstance 242
    - ElementBlock1D 257
    - ElementBlock2D 259, 260
    - ElementBlock3D 261
    - GenesisEntity 297, 298
- ElementBlock1D**
  - Base Class*
    - ElementBlock
  - Class Declaration* 257
  - Used In*
    - ElementBlock 253, 255
- ElementBlock2D**
  - Base Class*
    - ElementBlock
  - Class Declaration* 259
  - Used In*
    - ElementBlock 253, 255
- ElementBlock3D**
  - Base Class*
    - ElementBlock
  - Class Declaration* 261
  - Used In*
    - ElementBlock 253, 255
- ElementQuality**
  - Class Declaration* 263
  - Derived Classes*
    - HexQuality 341
    - QuadQuality 447
- ENTITY**
  - Used In*
    - AcisGeometryEngine 61
    - GeometryTool 316
    - Model 401
    - parents.hpp 421
    - RefBody 463
    - RefEdge 469
    - RefVolume 506
- ENTITY\_LIST**
  - Used In*
    - AcisGeometryEngine 61
    - GeometryCommands 304
    - GeometryTool 316
    - Model 401
    - parents.hpp 421
- ExodusMesh**
  - Base Class*
    - RefEntity
  - Class Declaration* 265
  - Used In*
    - AcisGeometryEngine 58, 61
    - GridSearch 324, 328
    - RefEntity 471, 473, 474, 475, 476
    - RefFace 488, 491, 493
    - RefVolume 506
- F**

---
- FACE**
  - Used In*
    - AcisGeometryEngine 61
    - Model 401
    - RefFace 493
- FaceUse**
  - Base Class*
    - CubitEntity
  - Class Declaration* 273
  - Derived Classes*
    - ProtoHex 437
  - Used In*

# Class Index

- CubitFace 139, 141, 142
- CubitHex 143, 144, 145
- CubitKnife 147, 148, 149
- CubitNode 166
- DrawingTool 222, 225
- DrawingToolInstance 239, 240, 242
- FullHex 289, 290, 291
- Hexer 333, 337
- NodeHex 412, 413
- ProtoHex 437, 438, 439, 441
- SortedNodeDLLList\* 87
- FastqBC**
  - Base Class*
    - FastqEntity
  - Class Declaration* 279
  - Used In*
    - FastqCommands 283, 284
    - FastqEntity 277
    - FastqSide 280
- FastqCommands**
  - Base Class*
    - CommandHandler
  - Class Declaration* 283
- FastqEntity**
  - Class Declaration* 277
  - Derived Classes*
    - FastqBC 279
    - FastqLine 278
    - FastqPoint 278
    - FastqRegion 279
    - FastqSide 280
  - Used In*
    - FastqSide 280
- FastqLine**
  - Base Class*
    - FastqEntity
  - Class Declaration* 278
  - Used In*
    - FastqCommands 283, 284
    - FastqEntity 277
    - FastqSide 280
- FastqModel**
  - Class Declaration* 277
- FastqPoint**
  - Base Class*
    - FastqEntity
- FastqRegion**
  - Class Declaration* 278
  - Used In*
    - FastqCommands 283, 284
    - FastqEntity 277
    - FastqSide 280
- FastqSide**
  - Base Class*
    - FastqEntity
  - Class Declaration* 280
  - Used In*
    - FastqCommands 283, 284
    - FastqEntity 277
    - FastqSide 280
- FileCommands**
  - Base Class*
    - CommandHandler
  - Class Declaration* 285
- FREDTool**
  - Base Class*
    - Tool
  - Class Declaration* 287
  - Used In*
    - GenesisTool 299, 302
- FullHex**
  - Base Class*
    - CubitHex
  - Class Declaration* 289
  - Used In*
    - CubitHex 144, 145
    - DrawingTool 222, 225
    - DrawingToolInstance 240, 242
    - Hexer 333, 334, 337
    - SortedNodeDLLList\* 87
    - WhiskerHex 728, 734
- FunctionData**
  - Used In*
    - AttributeCommands 77

# Class Index

CommandHandler 116  
FastqCommands 284  
FileCommands 285  
GenesisCommands 295  
GeometryCommands 304  
GraphicsCommands 322  
MeshingCommands 389  
ModelCommands 403  
QualityCommands 451

## G

---

### GenesisCommands

*Base Class*  
CommandHandler  
*Class Declaration* 295  
*Used In*  
FREDTool 288  
GenesisTool 299, 302  
XpatchTool 766

### GenesisEntity

*Base Class*  
CubitEntity  
*Class Declaration* 297  
*Derived Classes*  
ElementBlock 253  
NodeSet 415  
SideSet 541  
*Used In*  
FastqCommands 284  
function\_templates.hpp 293  
GenesisCommands 295  
GenesisTool 300, 302  
Model 397, 401  
SideSet 541, 543

### GenesisTool

*Base Class*  
Tool  
*Class Declaration* 299  
*Used In*  
ElementBlock 253, 255  
ElementBlock1D 257  
ElementBlock2D 259, 260  
ElementBlock3D 261  
NodeSet 415, 417  
SideSet 541, 543

### GeometryCommands

*Base Class*  
CommandHandler  
*Class Declaration* 303  
**GeometrySizingTool**  
*Base Class*  
EdgeMeshTool  
*Class Declaration* 307  
**GeometryTool**  
*Base Class*  
Tool  
*Class Declaration* 309  
*Used In*  
AcisGeometryEngine 56, 61  
MappingFace 375  
TDCopied 601  
TDProjection 631

### GetLongOpt

*Class Declaration* 319  
*Used In*  
UserInterface 682, 683

### GraphicsCommands

*Base Class*  
CommandHandler  
*Class Declaration* 321  
*Used In*  
DrawingTool 224, 225  
DrawingToolInstance 241, 242

### GridSearch

*Class Declaration* 323  
*Used In*  
BaseHexer 82  
ExodusMesh 265, 269  
PyramidTool 445  
SortedNodeDLLList\* 87  
SubMapTool 569, 570, 572  
TDCellIndex 597  
TDDistance 603

## H

---

### Hexer

*Base Class*  
BaseHexer  
*Class Declaration* 329  
*Used In*  
CubitKnife 147, 148, 149  
HexToVoid 347, 350

# Class Index

ProtoHex 437, 441  
RefVolume 506  
WhiskerFace 724, 725

## HexKnowingNode

*Base Class*  
PillowNode  
*Class Declaration* 339

## HexQuality

*Base Class*  
ElementQuality  
*Class Declaration* 341

## HexQualityController

*Base Class*  
QualityController  
*Class Declaration* 343

## HexTool

*Base Class*  
VolMeshTool  
*Class Declaration* 345  
*Derived Classes*  
BaseHexer 81  
DoubletPillower 206  
TwoHalfDimTool 675  
WhiskerWeaver 753  
*Used In*  
DoubletPillower 208  
SurfMeshTool 579, 582  
ToolDataUser 663  
TwoHalfDimTool 675, 679

## HexToVoid

*Base Class*  
BaseHexer  
*Class Declaration* 347  
*Used In*  
RefVolume 501, 506

## HitPoint

*Structure Declaration* 391

## I

## IntervalLinearProgram

*Base Class*  
LinearProgram  
*Class Declaration* 351  
*Used In*  
MappingFace 374, 375, 376  
TDLPEdge 615, 616

## istream

*Used In*  
CubitString 184

## L

## LinearProgram

*Class Declaration* 357  
*Derived Classes*  
IntervalLinearProgram 351  
*Used In*  
IntervalLinearProgram 352, 353, 355

## LOOP

*Used In*  
AcisGeometryEngine 61  
BoundaryLayerEdge 97  
BoundaryLayerFace 100  
BoundaryLayerTool 102  
GeometryTool 316  
MappingFace 376  
RefFace 493  
RefVolume 506

## Loop

*Base Class*  
DLCubitNodeList  
*Class Declaration* 363  
*Used In*  
CleanUp 109, 110, 113  
DynLoopList 247  
Paver 424, 425, 426, 427, 428, 429  
RefFace 493  
SurfMeshTool 580, 581, 582  
SurfSubDomain 587, 588

## LoopCollapser

*Base Class*  
SelfCrossingLoop  
*Class Declaration* 365  
*Used In*  
SelfCrossing 515

## LoopInterval

*Class Declaration* 369  
*Used In*  
SurfMapTool 576, 577, 578

## LoopJoiner

*Base Class*  
SelfCrossingLoop  
*Class Declaration* 371

# Class Index

## LUMP

### *Used In*

AcisGeometryEngine 61  
Model 401  
RefVolume 506

## M

---

## MappingFace

### *Base Class*

ToolData

### *Class Declaration* 373

### *Used In*

SurfMapTool 575, 576, 577, 578  
ToolData 659, 660

## MapToolSupport

### *Class Declaration* 377

### *Derived Classes*

SubMapTool 569  
SurfMapTool 575  
VolMapTool 687  
VolSubMapTool 693

## MemoryBlock

### *Class Declaration* 381

### *Used In*

MemoryAllocation.hpp 379, 380  
MemoryManager 383, 385

## MemoryManager

### *Class Declaration* 383

### *Used In*

ArrayMemory 64  
ControlPoint 119  
CubitEdge 133, 134  
CubitFace 139, 142  
CubitNode 161, 166  
DLLList 202, 204  
DynamicArray 246  
EdgeUse 251, 252  
FaceUse 273, 275  
FullHex 289, 291  
GridSearch 328  
HexQuality 341  
MappingFace 373, 376  
MemoryBlock 381, 382  
NodeHex 411, 413  
ProtoHex 437, 441  
QuadQuality 447, 448

## Queue

Queue 459  
QueueNode 457  
SDLList 513  
TDCellIndex 597  
TDCenterNode 599  
TDCopied 601  
TDDistance 603  
TDGenesisID 605  
TDGeometrySizing 607, 608  
TDHexKnowing 609

TDLaplace 611, 612  
TDLayer 613, 614  
TDLPEdge 615, 616  
TDMorph 619, 620  
TDNodeHardLine 621, 622  
TDNormal 623, 624  
TDPaver 625, 626  
TDPillow 627, 628  
TDProjection 631  
TDSizing 633  
TDStrider 635, 636  
TDSubMap 637, 641  
TDTipton 643, 644  
TDUVSpace 647  
TDWeight 651  
Tree 667, 669  
WhiskerFace 723, 725  
WhiskerHex 727, 734

## MeshEntity

### *Base Class*

CubitEntity  
ToolDataUser

### *Class Declaration* 387

### *Derived Classes*

BoundaryLayer 93  
CubitEdge 133  
CubitFace 139  
CubitHex 143  
CubitKnife 147  
CubitNode 161  
CubitSheet 179  
SheetChord 519  
WhiskerChord 713  
WhiskerHex 727  
WhiskerSheet 739

### *Used In*

# Class Index

- ElementQuality 263
- FaceUse 273, 275
- function\_templates.hpp 293
- HexQuality 341, 342
- QuadQuality 447, 448
- QualitySummary 455
- RefEdge 466, 470
- SheetChord 521, 524
- SheetChordPoint 525, 527, 528, 529, 530, 534
- WhiskerHex 731, 734
- WhiskerSheet 743, 752
- WhiskerWeaver 758, 762
- MeshingCommands**
  - Base Class*
  - CommandHandler
  - Class Declaration* 389
  - Used In*
    - GenesisTool 299, 302
- MeshIntersect**
  - Class Declaration* 391
  - Used In*
    - BaseHexer 82
    - SortedNodeDLLList\* 87
- MeshTool**
  - Base Class*
  - Tool
  - Class Declaration* 395
  - Derived Classes*
    - EdgeMeshTool 249
    - ParallelMeshTool 419
    - SurfMeshTool 579
    - VolMeshTool 691
  - Used In*
    - RefVolume 501, 504, 505, 506
- MessageFlag**
  - Class Declaration* 155
  - Used In*
    - CubitMessage 155, 157
- Model**
  - Class Declaration* 397
  - Used In*
    - AcisGeometryEngine 57, 61
    - ExodusMesh 267, 269
    - GeometryTool 309, 316
    - MapToolSupport 378
- RefEntity 474, 476
- ModelCommands**
  - Base Class*
  - CommandHandler
  - Class Declaration* 403
- ModelViewer**
  - Class Declaration* 405
- Mouse**
  - Class Declaration* 407
  - Used In*
    - Mouse 408
    - Mouse.hpp 407
- MuseOutput**
  - Base Class*
  - Tool
  - Class Declaration* 409
- N**

---
- NodeHex**
  - Base Class*
  - CubitHex
  - Class Declaration* 411
  - Used In*
    - CubitHex 144, 145
    - DrawingTool 222, 225
    - DrawingToolInstance 240, 242
    - WhiskerHex 728, 734
- NodeSet**
  - Base Class*
  - GenesisEntity
  - Class Declaration* 415
  - Used In*
    - DrawingTool 222, 225
    - DrawingToolInstance 240, 242
    - GenesisEntity 297, 298
- O**

---
- ofstream**
  - Used In*
    - CubitMessage 157
- ostream**
  - Used In*
    - CubitString 184
    - CubitVector 186
- ostrstream**
  - Used In*

# Class Index

- CommandHandler 116
- outcome**  
*Used In*  
AcisGeometryEngine 61
- 
- P**
- par\_pos**  
*Used In*  
RefFace 493
- ParallelMeshTool**  
*Base Class*  
MeshTool  
*Class Declaration* 419
- Paver**  
*Base Class*  
SurfMeshTool  
*Class Declaration* 423  
*Used In*  
ProtoHex 439, 440, 441  
RefFace 484, 493  
SurfMeshTool 582
- PillowNode**  
*Class Declaration* 431  
*Derived Classes*  
HexKnowingNode 339  
*Used In*  
HexKnowingNode 339  
StarNode 555
- PillowSheet**  
*Base Class*  
WhiskerSheet  
*Class Declaration* 433  
*Used In*  
WhiskerSheet 741, 752
- position**  
*Used In*  
AcisGeometryEngine 61  
RefEdge 470  
RefFace 493  
RefVertex 499
- ProtoHex**  
*Base Class*  
FaceUse  
*Class Declaration* 437  
*Used In*  
BaseHexer 81
- CubitKnife 147, 148, 149  
FaceUse 273, 275  
Hexer 329, 330, 331, 332, 333, 337  
HexToVoid 347, 348, 349, 350  
SortedNodeDLLList\* 82, 83, 84, 85, 86, 87
- Pyramid**  
*Class Declaration* 443  
*Used In*  
PyramidTool 445  
RefVolume 506
- PyramidTool**  
*Base Class*  
VolMeshTool  
*Class Declaration* 445  
*Used In*  
RefVolume 501, 506
- 
- Q**
- QuadQuality**  
*Base Class*  
ElementQuality  
*Class Declaration* 447
- QuadQualityController**  
*Base Class*  
QualityController  
*Class Declaration* 449
- QualityCommands**  
*Base Class*  
CommandHandler  
*Class Declaration* 451  
*Used In*  
GenesisTool 299, 302
- QualityController**  
*Class Declaration* 453  
*Derived Classes*  
HexQualityController 343  
QuadQualityController 449  
*Used In*  
RefEntity 471, 475, 476
- QualitySummary**  
*Class Declaration* 455  
*Used In*  
QualityController 453
- Queue**  
*Class Declaration* 458

# Class Index

## *Used In*

Queue 459  
QueueNode 457

## **QueueNode**

### *Class Declaration* 457

### *Used In*

Queue 457, 458, 459

---

## R

## **RefBody**

### *Base Class*

RefEntity

### *Class Declaration* 461

### *Used In*

AcisGeometryEngine 57, 58, 59, 61  
CopiedData 121  
DrawingTool 217, 218, 224, 225  
DrawingToolInstance 241, 242  
FastqCommands 284  
GeometryCommands 303, 304  
GeometryTool 310, 311, 312, 313, 314, 315, 316  
Model 397, 398, 401  
RefEdge 466, 470  
RefEntity 474, 475, 476  
RefFace 485, 493  
RefVertex 497, 499  
RefVolume 505, 506  
StairTool 551, 552, 553

## **RefEdge**

### *Base Class*

RefEntity

ToolDataUser

### *Class Declaration* 465

### *Used In*

AcisGeometryEngine 58, 59, 60, 61  
BoundaryLayer 94  
BoundaryLayerEdge 95, 97  
BoundaryLayerFace 99, 100  
CleanUp 113  
ControlPoint 119, 120  
CubitNode 165, 166  
CurveSubDomain 195, 196  
DrawingTool 217, 218, 222, 224, 225  
DrawingToolInstance 239, 241, 242  
EdgeMeshTool 249, 250

ElementBlock1D 257, 258

ExodusMesh 266, 269

FastqCommands 283, 284

FastqLine 278

FastqSide 280

GeometrySizingTool 307

GeometryTool 311, 314, 315, 316

IntervalLinearProgram 351, 352, 353, 354, 355

MappingFace 373, 375, 376

MapToolSupport 378

Model 397, 398, 401

ModelViewer 405, 406

NodeSet 415, 416, 417

RefEntity 474, 476

RefFace 484, 485, 489, 490, 492, 493

RefVolume 505, 506

RotateTool 510

SideSet 541, 543

Strider 561

SubDomain 563

SurfMapTool 576, 578

SurfMeshTool 582

SurfMorphTool 585, 586

SurfSubDomain 588

TDGeometrySizing 607, 608

TDLPEdge 615, 616

TranslateTool 666

TwoHalfDimTool 678, 679

VolSubMapTool 694, 695, 697

VolumeEdgeType 699

## **RefEntity**

### *Base Class*

CubitEntity

### *Class Declaration* 471

### *Derived Classes*

ExodusMesh 265

RefBody 461

RefEdge 465

RefFace 483

RefGroup 495

RefVertex 497

RefVolume 501

SubDomain 563

### *Used In*

AcisGeometryEngine 55, 56, 57, 59,

# Class Index

- 60, 61
- ATTRIB\_CUBIT\_OWNER** 67
- AttributeCommands** 77
- CopiedData** 121
- CubitEdge** 133, 134
- CubitEntity** 138
- CubitFace** 139, 141, 142
- CubitHex** 143, 144, 145
- CubitNode** 161, 162, 163, 164, 166
- EdgeUse** 251, 252
- ElementBlock** 253, 255
- ElementBlock1D** 257, 258
- ElementBlock2D** 259, 260
- ElementBlock3D** 261
- ExodusMesh** 265, 266, 267, 268, 269
- FaceUse** 273, 275
- FullHex** 289, 290, 291
- function\_templates.hpp** 293
- GenesisEntity** 297, 298
- GenesisTool** 299, 302
- GeometryCommands** 304
- GeometrySizingTool** 307
- GeometryTool** 311, 312, 315, 316
- Hexer** 329, 331, 337
- HexQuality** 342
- HexQualityController** 343
- MeshEntity** 387
- Model** 397, 401
- ModelViewer** 405, 406
- NodeHex** 411, 412, 413
- NodeSet** 415, 417
- ProtoHex** 437, 441
- QuadQuality** 448
- QuadQualityController** 449
- QualityCommands** 451
- QualityController** 453
- RefBody** 463
- RefEdge** 469, 470
- RefEntityName** 479, 480
- RefEntityNameMap** 481
- RefFace** 490, 493
- RefGroup** 496
- RefVertex** 499
- RefVolume** 505, 506
- SideSet** 541, 543
- SizingTool** 545
- SubMapNode** 565, 566
- TDSubMap** 638, 641
- WhiskerFace** 725
- WhiskerHex** 730, 734
- RefEntityName**
  - Class Declaration* 479
- RefEntityNameMap**
  - Class Declaration* 481
  - Used In*
    - RefEntityName** 479, 480
- RefEntityNameMapList**
  - Used In*
    - RefEntityName** 479, 480
- RefFace**
  - Base Class*
  - RefEntity**
  - ToolDataUser**
  - Class Declaration* 483
  - Used In*
    - AcisGeometryEngine** 57, 58, 59, 60, 61
    - AutoVertexType** 79
    - BoundaryLayer** 94
    - BoundaryLayerFace** 99, 100
    - BoundaryLayerTool** 102
    - CleanUp** 112, 113
    - CubitFace** 140, 142
    - CubitKnife** 148, 149
    - CubitNode** 165, 166
    - CubitVoxel** 192, 193
    - DrawingTool** 217, 218, 222, 224, 225
    - DrawingToolInstance** 239, 241, 242
    - ElementBlock2D** 259, 260
    - ExodusMesh** 267, 269
    - FastqCommands** 283, 284
    - FastqRegion** 279
    - FastqSide** 280
    - GeometryCommands** 304, 305
    - GeometryTool** 310, 311, 312, 314, 315, 316
    - GridSearch** 324, 328
    - HexToVoid** 347, 350
    - MappingFace** 373, 374, 375, 376
    - MapToolSupport** 378
    - Model** 397, 398, 401
    - ModelViewer** 405, 406
    - NodeHex** 413

# Class Index

- NodeSet 415, 416, 417
- ParallelMeshTool 419, 420
- Paver 428, 429
- ProtoHex 437, 438, 440, 441
- RefEdge 466, 468, 470
- RefEntity 472, 474, 476
- RefVolume 501, 502, 503, 504, 506
- RotateTool 509, 510
- SideSet 541, 543
- SortedNodeDLLList\* 83, 88
- Strider 561
- SubDomain 563
- SubMapNode 566
- SubMapTool 570, 572
- SurfDistributeTool 573
- SurfMapTool 575, 577, 578
- SurfMeshTool 580, 582
- SurfMorphTool 585, 586
- SurfSubDomain 587, 588
- SweepTool 595, 596
- TDNormal 623, 624
- TDStrider 635, 636
- TDSubMap 639, 641
- TranslateTool 665, 666
- TriangleTool 671, 672
- TriElementTool 673
- TwoHalfDimTool 675, 679
- VolMapTool 687, 688
- VolSubMapTool 695, 697
- VoxelTool 705, 706
- RefGroup**
  - Base Class*
  - RefEntity
  - Class Declaration* 495
  - Used In*
    - Model 397, 398, 401
    - NodeSet 415, 417
    - RefEntity 474, 475, 476
    - SideSet 541, 543
- RefVertex**
  - Base Class*
  - RefEntity
  - Class Declaration* 497
  - Used In*
    - AcisGeometryEngine 58, 59, 60, 61
    - AutoVertexType 79
- BoundaryLayerEdge 95, 96, 97
- BoundaryLayerTool 102
- CubitBox 125, 126
- CubitNode 165, 166
- CubitVector 186
- CurveSubDomain 196
- DrawingTool 217, 218, 222, 224, 225
- DrawingToolInstance 230, 239, 241, 242
- EdgeMeshTool 249, 250
- FastqCommands 283, 284
- FastqPoint 278
- FastqSide 280
- GeometryTool 311, 313, 314, 315, 316
- GraphicsCommands 321, 322
- MappingFace 373, 376
- Model 397, 398, 401
- NodeSet 415, 416, 417
- RefEdge 465, 467, 468, 470
- RefEntity 474, 477
- RefFace 484, 485, 486, 488, 489, 492, 493
- SurfMorphTool 585, 586
- SurfVertexType 593, 594

## RefVolume

- Base Class*
- RefEntity
- Class Declaration* 501
- Used In*
  - AcisGeometryEngine 57, 58, 59, 60, 61
  - AutoVertexType 79
  - CubitEdge 134
  - CubitFace 140, 142
  - CubitHex 145
  - CubitNode 161, 166
  - DoubletPillower 208
  - DrawingTool 217, 218, 224, 225
  - DrawingToolInstance 241, 242
  - ElementBlock3D 261
  - FastqRegion 279
  - FastqSide 280
  - FullHex 291
  - GeometryTool 311, 312, 313, 315, 316
  - GridSearch 324, 325, 328
  - Hexer 329, 332, 337
  - HexTool 345, 346

# Class Index

HexToVoid 349, 350  
MapToolSupport 378  
Model 397, 398, 401  
NodeHex 413  
NodeSet 415, 416, 417  
ProtoHex 437, 441  
Pyramid 443  
PyramidTool 445  
RefBody 461, 462, 463  
RefEntity 474, 477  
RefFace 483, 486, 487, 488, 492, 493  
RotateTool 509, 510  
SheetChord 524  
SortedNodeDLLList\* 87, 88  
StairTool 551, 552, 553  
SubDomain 563  
SubMapTool 569, 571, 572  
SurfMapTool 575, 577, 578  
SurfMeshTool 579, 581, 582  
SweepTool 595, 596  
TranslateTool 665, 666  
TwoHalfDimTool 675, 679  
VolMapTool 687, 688  
VolMeshTool 691  
VolSubMapTool 696, 697  
VolVoidBoundary 701  
VolVoidSepTool 703, 704  
WhiskerChord 720  
WhiskerSheet 752  
WhiskerWeaver 753, 755, 760, 762

## RotateTool

*Base Class*  
TwoHalfDimTool  
*Class Declaration* 509

## S

---

## SDLControlPointList

*Used In*  
TDGeometrySizing 607, 608

## SDLList

*Base Class*  
DLLList  
*Class Declaration* 511  
*Used In*  
ArrayBasedContainer 64  
ArrayMemory 64

## SDLLoopIntervalList

*Used In*  
LoopInterval 369

## SelfCrossing

*Class Declaration* 515  
*Used In*  
LoopCollapser 365, 366  
LoopJoiner 371  
SelfCrossingLoop 517, 518

## SelfCrossingLoop

*Base Class*  
DLSelfCrossingList  
*Class Declaration* 517  
*Derived Classes*  
LoopCollapser 365  
LoopJoiner 371  
*Used In*  
LoopCollapser 365, 366  
LoopJoiner 371  
SelfCrossing 515

## SheetChord

*Base Class*  
MeshEntity  
*Class Declaration* 519  
*Used In*  
DrawingTool 222, 225  
DrawingToolInstance 240, 242  
MeshEntity 387  
PillowSheet 434, 435  
SheetChordPoint 527, 534  
SheetEdge 535, 536  
StcEdge 558, 559  
WhiskerChord 713, 714, 715, 716, 717,  
718, 719, 720  
WhiskerHex 728, 729, 734  
WhiskerKnife 735, 737  
WhiskerSheet 740, 742, 743, 744, 745,  
746, 747, 749, 751, 752  
WhiskerWeaver 756, 758, 759, 762

## SheetChordPoint

*Class Declaration* 525  
*Used In*  
BladeEdges 89, 90  
MuseOutput 409  
PillowSheet 433, 435  
SheetChord 520, 521, 522, 523, 524

# Class Index

- SheetEdge 535, 536
- StcEdge 557, 558, 559
- WhiskerCell 709, 710, 711, 712
- WhiskerChord 713, 714, 716, 720
- WhiskerFace 725
- WhiskerHex 727, 728, 729, 730, 731, 732, 733, 734
- WhiskerKnife 735, 737
- WhiskerSheet 740, 743, 744, 745, 747, 748, 752
- WhiskerWeaver 755, 757, 758, 759, 761, 762
- SheetEdge**
  - Class Declaration* 535
  - Used In*
    - BladeEdges 89, 90
    - SheetEdgeTree.hpp 537
    - StcEdge 557, 558, 559
    - WhiskerKnife 735, 736, 737
- SheetEdgeTree**
  - Used In*
    - SheetEdgeTree.hpp 537
    - SheetEdgeTreeQueue.hpp 539
- SheetEdgeTreeQueue**
  - Used In*
    - SheetEdgeTreeQueue.hpp 539
- SideSet**
  - Base Class*
    - GenesisEntity
  - Class Declaration* 541
  - Used In*
    - DrawingTool 222, 225
    - DrawingToolInstance 240, 242
    - GenesisEntity 298
- SizingTool**
  - Base Class*
    - Tool
  - Class Declaration* 545
- SLLList**
  - Base Class*
    - CubitCollection
  - Class Declaration* 548
  - Used In*
    - SLLListItem 547
    - SLLListIterator 549
- SLLListItem**
  - Class Declaration* 547
  - Used In*
    - SLLList 547, 548
    - SLLListItem 547
    - SLLListIterator 549
- SLLListIterator**
  - Class Declaration* 549
  - Used In*
    - SLLList 547, 548
    - SLLListItem 547
- SLRefEntityList**
  - Used In*
    - ElementBlock 253, 254, 255
- SLRefEntityListIterator**
  - Used In*
    - ElementBlock 253, 255
- SortedNodeDLLList\***
  - Structure Declaration* 82
- StairTool**
  - Base Class*
    - Tool
  - Class Declaration* 551
- StarNode**
  - Class Declaration* 555
  - Used In*
    - PillowNode 431, 432
    - TDPillow 627, 628
- StcEdge**
  - Class Declaration* 557
  - Used In*
    - WhiskerCell 711, 712
- Strider**
  - Base Class*
    - EdgeMeshTool
  - Class Declaration* 561
  - Used In*
    - GeometrySizingTool 307
- SubDomain**
  - Base Class*
    - RefEntity
  - Class Declaration* 563
  - Derived Classes*
    - CurveSubDomain 195
    - SurfSubDomain 587
- SubMapNode**
  - Base Class*

# Class Index

- ToolData  
        *Class Declaration* 565
- SubMapTool**
  - Base Class*
    - MapToolSupport
    - SurfMeshTool
  - Class Declaration* 569
- surface**
  - Used In*
    - AcisGeometryEngine 61
    - RefFace 493
- SurfDistributeTool**
  - Base Class*
    - SurfMeshTool
  - Class Declaration* 573
- SurfMapTool**
  - Base Class*
    - MapToolSupport
    - SurfMeshTool
  - Class Declaration* 575
- Used In*
  - LoopInterval 369
  - MappingFace 374, 376
  - VolMapTool 687, 688
- SurfMeshTool**
  - Base Class*
    - MeshTool
  - Class Declaration* 579
  - Derived Classes*
    - BoundaryLayerTool 101
    - CleanUp 105
    - Paver 423
    - SubMapTool 569
    - SurfDistributeTool 573
    - SurfMapTool 575
    - TriangleTool 671
    - TriElementTool 673
  - Used In*
    - CubitFace 139, 142
    - Paver 423, 429
    - RefFace 493
    - TNormal 623, 624
- SurfMorphTool**
  - Class Declaration* 585
- SurfSubDomain**
  - Base Class*
- SubDomain  
        *Class Declaration* 587
- SurfVertexTree**
  - Used In*
    - AutoVertexType 79
    - SurfVertexTree.hpp 591
- SurfVertexType**
  - Class Declaration* 593
  - Used In*
    - AutoVertexType 79
    - RefFace 489, 493
    - SubMapTool 571, 572
    - SurfVertexTree.hpp 591
- SweepTool**
  - Base Class*
    - TwoHalfDimTool
  - Class Declaration* 595
  - Used In*
    - TwoHalfDimTool 678, 679

## T

---

- TDCellIndex**
  - Base Class*
    - ToolData
  - Class Declaration* 597
  - Used In*
    - ToolData 659, 660
- TDCenterNode**
  - Base Class*
    - ToolData
  - Class Declaration* 599
  - Used In*
    - ToolData 659, 660
- TDCopied**
  - Base Class*
    - ToolData
  - Class Declaration* 601
  - Used In*
    - GeometryTool 315, 316
    - ToolData 659, 660
- TDDistance**
  - Base Class*
    - ToolData
  - Class Declaration* 603
  - Used In*
    - ToolData 659, 660

# Class Index

<b>TDGenesisID</b>	<i>Class Declaration</i> 619
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 605	
<i>Used In</i>	
ToolData 659, 660	
<b>TDGeometrySizing</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 607	
<i>Used In</i>	
ControlPoint 119, 120	
GeometrySizingTool 307	
Strider 561	
ToolData 659, 660	
<b>TDHexKnowing</b>	
<i>Base Class</i>	
TDPillow	
<i>Class Declaration</i> 609	
<i>Used In</i>	
DoubletPillowerTD 211, 213	
ToolData 659, 660	
<b>TDLaplace</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 611	
<i>Used In</i>	
HexTool 345, 346	
ToolData 659, 660	
<b>TDLayer</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 613	
<i>Used In</i>	
ToolData 659, 660	
<b>TDLPEdge</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 615	
<i>Used In</i>	
IntervalLinearProgram 351, 352, 353, 354, 355	
ToolData 659, 660	
<b>TDMorph</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 605	
<i>Used In</i>	
SurfMorphTool 585, 586	
ToolData 659, 660	
<b>TDNodeHardLine</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 621	
<i>Used In</i>	
Paver 427, 429	
ToolData 659, 660	
<b>TDNormal</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 623	
<i>Derived Classes</i>	
TDPaver 625	
<i>Used In</i>	
TDPaver 625, 626	
ToolData 659, 660	
<b>TDPaver</b>	
<i>Base Class</i>	
TDNormal	
TDSizing	
<i>Class Declaration</i> 625	
<i>Used In</i>	
MemoryAllocation.hpp 379, 380	
Paver 427, 428, 429	
TDNormal 623, 624	
ToolData 659, 660	
<b>TDPillow</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 627	
<i>Derived Classes</i>	
TDHexKnowing 609	
<i>Used In</i>	
DoubletPillowerTD 211, 212, 213	
StarNode 555	
TDHexKnowing 609	
ToolData 659, 660	
<b>TDProjection</b>	
<i>Base Class</i>	
ToolData	
<i>Class Declaration</i> 631	
<i>Used In</i>	

# Class Index

- ToolData 659, 660
- TwoHalfDimTool 677, 678, 679
- TDSizing**
  - Base Class*
    - ToolData
  - Class Declaration* 633
  - Derived Classes*
    - TDPaver 625
  - Used In*
    - ExodusMesh 267, 269
    - SurfMeshTool 580, 582
    - TDPaver 625, 626
    - ToolData 659, 660
- TDStrider**
  - Base Class*
    - ToolData
  - Class Declaration* 635
  - Used In*
    - Strider 561
    - ToolData 659, 660
- TDSubMap**
  - Base Class*
    - ToolData
  - Class Declaration* 637
  - Used In*
    - MapToolSupport 378
    - SubMapTool 569, 570, 571, 572
    - ToolData 659, 660
    - VolSubMapTool 693, 694, 695, 696, 697
- TDTipton**
  - Base Class*
    - ToolData
  - Class Declaration* 643
  - Used In*
    - HexTool 345, 346
    - SurfMeshTool 579, 582
    - ToolData 659, 660
- TDUVSpace**
  - Base Class*
    - ToolData
  - Class Declaration* 647
  - Used In*
    - RefFace 489, 491, 493
    - SurfMapTool 577, 578
    - ToolData 659, 660
- TDVolMap**
  - Base Class*
    - ToolData
  - Class Declaration* 649
- TDWeight**
  - Base Class*
    - ToolData
  - Class Declaration* 651
  - Used In*
    - HexTool 345, 346
    - ToolData 659, 660
- TimeVal**
  - Class Declaration* 653
  - Derived Classes*
    - AbsTime 654
    - DeltaTime 655
  - Used In*
    - AbsTime 654
    - DeltaTime 655
- Tool**
  - Class Declaration* 657
  - Derived Classes*
    - DrawingTool 215
    - DrawingToolInstance 227
    - FREDTool 287
    - GenesisTool 299
    - GeometryTool 309
    - MeshTool 395
    - MuseOutput 409
    - SizingTool 545
    - StairTool 551
    - VoxelTool 705
    - XpatchTool 765
  - Used In*
    - GeometryCommands 303, 304, 305
    - GeometryTool 310, 311, 316
    - WhiskerSheet 739, 752
- ToolData**
  - Class Declaration* 659
  - Derived Classes*
    - MappingFace 373
    - SubMapNode 565
    - TDCellIndex 597
    - TDCenterNode 599
    - TDCopied 601
    - TDDistance 603

# Class Index

TDGenesisID 605  
TDGeometrySizing 607  
TDLaplace 611  
TDLayer 613  
TDLPEdge 615  
TDMorph 619  
TDNodeHardLine 621  
TDNormal 623  
TDPillow 627  
TDPProjection 631  
TDSizing 633  
TDStrider 635  
TDSubMap 637  
TDTipton 643  
TDUVSpace 647  
TDVolMap 649  
TDWeight 651  
WhiskerFace 723  
WhiskerHex 727

## Used In

CleanUp 112, 113  
CubitNode 161, 166  
DoubletPillowerTD 211, 213  
GeometrySizingTool 307  
IntervalLinearProgram 352, 354, 355  
MappingFace 374, 376  
RefEdge 465, 470  
SubMapNode 565, 566  
SurfMapTool 577, 578  
TDCellIndex 597  
TDCenterNode 599  
TDCopied 601  
TDDistance 603  
TDGenesisID 605  
TDGeometrySizing 608  
TDLaplace 611, 612  
TDLayer 613, 614  
TDLPEdge 615, 616  
TDMorph 619, 620  
TDNormal 623, 624  
TDPillow 628  
TDPProjection 631  
TDSizing 633  
TDStrider 635, 636  
TDSubMap 638, 641  
TDTipton 644

TDUVSpace 647  
TDWeight 651  
ToolDataUser 663  
WhiskerFace 723, 725

## ToolDataUser

*Class Declaration 663*  
*Derived Classes*  
MeshEntity 387  
RefEdge 465  
RefFace 483

## TranslateTool

*Base Class*  
TwoHalfDimTool  
*Class Declaration 665*

## Tree

*Class Declaration 667*  
*Used In*  
CubitDefines 131  
CubitEntity 137, 138  
GenesisEntity 297, 298  
MeshEntity 387  
RefEntity 471, 477

## TriangleTool

*Base Class*  
SurfMeshTool  
*Class Declaration 671*

## TriElementTool

*Base Class*  
SurfMeshTool  
*Class Declaration 673*  
*Used In*  
RefFace 489, 493

## TwoHalfDimTool

*Base Class*  
HexTool  
*Class Declaration 675*  
*Derived Classes*  
RotateTool 509  
SweepTool 595  
TranslateTool 665  
*Used In*  
RotateTool 509, 510  
TDLayer 613, 614  
TDPProjection 631  
TranslateTool 665, 666

# Class Index

## U

---

### UserInterface

*Class Declaration* 681

## V

---

### VERTEX

*Used In*

- AcisGeometryEngine 61
- Model 401
- RefFace 493
- RefVertex 499

### VolMapTool

*Base Class*

- MapToolSupport
- VolMeshTool

*Class Declaration* 687

*Used In*

- CubitNode 161, 166
- MapToolSupport 377, 378
- TDVolMap 649
- TDVolMapTool 649

### VolMeshTool

*Base Class*

- MeshTool

*Class Declaration* 691

*Derived Classes*

- HexTool 345
- PyramidTool 445
- VolMapTool 687
- VolSubMapTool 693
- VolVoidSepTool 703

*Used In*

- VolMapTool 688

### VolSubMapTool

*Base Class*

- MapToolSupport
- VolMeshTool

*Class Declaration* 693

*Used In*

- SubMapTool 571, 572
- TDSubMap 637, 641

### VolumeEdgeType

*Class Declaration* 699

*Used In*

- RefVolume 505, 506

## VolVoidBoundary

*Base Class*

- DLCubitFaceList

*Class Declaration* 701

*Used In*

- VolVoidSepTool 703, 704

### VolVoidSepTool

*Base Class*

- VolMeshTool

*Class Declaration* 703

### VoxelTool

*Base Class*

- Tool

*Class Declaration* 705

*Used In*

- CubitVoxel 191, 193

- SurfDistributeTool 573

## W

---

### WhiskerCell

*Class Declaration* 709

*Used In*

- SheetChordPoint 527, 534

- StcEdge 557, 558, 559

- WhiskerHex 730, 734

- WhiskerWeaver 759, 762

### WhiskerChord

*Base Class*

- MeshEntity

*Class Declaration* 713

*Used In*

- DrawingTool 222, 225

- DrawingToolInstance 240, 242

- PillowSheet 433, 434, 435

- SheetChord 519, 523, 524

- WhiskerFace 723, 725

- WhiskerHex 727, 728, 729, 730, 731, 733, 734

- WhiskerKnife 735, 736, 737

- WhiskerSheet 740, 744, 752

- WhiskerWeaver 757, 762

### WhiskerFace

*Base Class*

- ToolData

*Class Declaration* 723

*Used In*

# Class Index

SheetChord 524

SheetChordPoint 527, 534

ToolData 659, 661

WhiskerChord 713, 714, 716, 718, 720

WhiskerHex 729, 734

WhiskerSheet 742, 744, 745, 752

WhiskerWeaver 762

## WhiskerHex

*Base Class*

MeshEntity

ToolData

*Class Declaration* 727

*Used In*

MeshEntity 387

Model 398, 400, 401

PillowSheet 433, 435

SheetChord 522, 524

SheetChordPoint 525, 534

SortedNodeDLLList\* 87, 88

ToolData 659, 661

WhiskerCell 709, 710, 711, 712

WhiskerChord 714, 715, 716, 717, 718,  
719, 720

WhiskerFace 725

WhiskerWeaver 757, 758, 759, 762

## WhiskerKnife

*Class Declaration* 735

*Used In*

BladeEdges 89, 90

WhiskerSheet 746, 752

WhiskerWeaver 753, 762

## WhiskerSheet

*Base Class*

MeshEntity

*Class Declaration* 739

*Derived Classes*

PillowSheet 433

*Used In*

CubitKnife 147, 148, 149

DrawingTool 222, 225

DrawingToolInstance 240, 242

Model 398, 400, 401

MuseOutput 409

PillowSheet 433, 434, 435

RefEntity 477

SheetChord 519, 520, 524

SheetChordPoint 525, 526, 528, 529,

530, 534

WhiskerCell 709, 712

WhiskerChord 713, 714, 715, 717, 719,  
720

WhiskerFace 723, 724, 725

WhiskerHex 728, 729, 734

WhiskerWeaver 756, 757, 760, 762

## WhiskerWeaver

*Base Class*

HexTool

*Class Declaration* 753

*Used In*

PillowSheet 434, 435

RefVolume 501, 503, 506

SheetChord 521, 524

WhiskerSheet 739, 741, 751, 752

## WIRE

*Used In*

AcisGeometryEngine 61

## X

---

## XpatchTool

*Base Class*

Tool

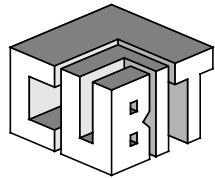
*Class Declaration* 765

*Used In*

GenesisTool 299, 302

# Class Index

# Class Index



---

# Function Index

(Protected) 245  
(Public) 167, 169, 170, 175, 176, 177  
- (Public) 153, 653, 654, 655  
- (Public, friend) 126

## Symbols

---

701  
!= (Public) 183  
& (Public, friend) 126  
&= (Public) 126  
\* (Public) 153  
\* (Public, friend) 126  
\*= (Public) 126  
+ (Public) 654, 655  
+ (Public, friend) 126, 183  
+= (Public) 63, 126, 183, 654, 655  
/ (Public, friend) 126  
/= (Public) 126  
= (Public) 63, 126, 153, 167, 172, 183, 200, 511  
-= (Public) 126, 654, 655  
== (Public) 183  
\_3d\_segment() (Public) 221, 238  
| (Public, friend) 126  
|= (Public) 126

## A

---

abort() (Private) 428, 585  
about\_spatially\_equal() (Public) 55, 313, 314  
AbsTime (Private, constructor) 654  
AbsTime (Public, constructor) 654  
access\_voids() (Public) 703  
ACIS\_API\_error() (Private) 315  
ACIS\_API\_Error() (Public) 60  
acis\_attribute\_exists() (Public) 59  
acis\_curves() (Public) 468, 488  
acis\_debug\_body() (Public) 60

acis\_debug\_coedge() (Public) 60  
acis\_debug\_dump() (Public) 490  
acis\_debug\_edge() (Public) 60  
acis\_debug\_face() (Public) 60  
acis\_debug\_loop() (Public) 60  
acis\_version() (Public, static) 55  
AcisGeometryEngine (Protected, constructor) 55  
AcisGeometryEngine (Public, destructor) 55  
act\_on\_intersect() (Private) 329, 347  
act\_on\_intersect() (Public, virtual) 82  
act\_on\_perturb\_bdry() (Private) 329, 347  
act\_on\_perturb\_bdry() (Public, virtual) 82  
act\_on\_seam\_case\_5() (Private) 329, 347  
act\_on\_seam\_case\_5() (Public, virtual) 82  
act\_on\_unusable() (Private) 329, 347  
act\_on\_unusable() (Public, virtual) 82  
activate() (Public) 737  
add() (Public) 222, 239, 240, 253, 398, 415, 541, 588  
add() (Public, virtual) 297  
add\_a\_face() (Private) 424  
add\_blind\_sheet\_chord() (Public) 742  
add\_boundary\_edge() (Public) 334  
add\_boundary\_face() (Public) 334  
add\_boundary\_node() (Public) 334  
add\_bounding\_face() (Public) 504  
add\_child\_item() (Public) 668  
add\_chord\_point() (Public) 520  
add\_constraint() (Public) 357  
add\_count() (Private) 576  
add\_crossing() (Protected) 517  
add\_doublet\_points() (Public) 755  
add\_dummy\_variable() (Public) 351  
add\_edge() (Private) 231  
add\_edge() (Public) 163, 192, 268, 467, 487, 504, 706  
add\_edge() (Public, virtual) 473

# Function Index

add\_edge\_use() (Public) 133  
add\_edges() (Private) 232  
add\_face() (Public) 179, 251, 268, 487, 504, 517  
add\_face() (Public, virtual) 473  
add\_face\_sheet() (Public) 724  
add\_face\_use() (Public) 141  
add\_hard\_point() (Public) 489  
add\_hex() (Public) 274, 504, 714  
add\_hex() (Public, virtual) 472  
add\_higher\_order\_nodes() (Public) 467, 487, 504  
add\_if\_surface\_doublet() (Private) 206  
add\_input\_file() (Public) 681  
add\_inside\_face() (Public) 742  
add\_inside\_sheet\_chord() (Public) 743  
add\_internal\_face() (Public) 334  
add\_journal\_entry() (Protected, static) 115  
add\_knife() (Public) 274  
add\_neighbor() (Public) 526  
add\_no\_test\_edge() (Public) 625  
add\_node() (Private) 232  
add\_node() (Public) 192, 268, 324, 467, 487, 498, 504, 622, 706, 709, 730  
add\_node() (Public, virtual) 473  
add\_node\_to\_cell() (Private) 324  
add\_on\_corner() (Private) 424  
add\_on\_reversal() (Private) 424  
add\_on\_side() (Private) 424  
add\_parent() (Public) 73  
add\_point() (Public) 731  
add\_quad\_to\_facet\_list() (Private) 231  
add\_ref\_edge() (Public) 278  
add\_ref\_entity() (Public) 496  
add\_ref\_face() (Public) 504  
add\_ref\_vertex() (Public) 278  
add\_refedge() (Public) 99  
add\_refentity\_name() (Public) 479  
add\_row() (Private) 424  
add\_sample() (Public) 455  
add\_sheet() (Public) 398  
add\_sheet\_chord() (Public) 742, 756  
add\_simple\_neighbors\_unique() (Private) 703  
add\_sizing\_function() (Public) 268  
add\_TD() (Public) 663  
add\_temp() (Public) 351  
add\_third\_neighbor() (Public) 526  
add\_to\_denominator() (Public) 730  
add\_to\_error\_count() (Private) 155  
add\_to\_group\_list() (Public) 475  
add\_to\_interior\_loop() (Private) 101  
add\_to\_numerator() (Public) 730  
add\_to\_star\_list() (Private) 206  
add\_to\_star\_list\_if\_wedgeless() (Private) 206  
add\_triangle\_face() (Public) 747  
add\_variable() (Public) 357  
add\_whisker\_face\_to\_loop() (Public) 741  
add\_whisker\_hex() (Public) 398  
add\_whisker\_sheet() (Public) 398, 760  
addRefVolume() (Public) 462  
adjacent\_nodes() (Public) 412  
adjoining\_face() (Public) 468  
adjoins() (Public) 490  
adjoint() (Public) 153  
adjust\_face\_uses() (Private) 289  
adjust\_join\_points() (Public) 521  
adjust\_join\_points\_and\_states() (Public) 716  
adjust\_loop() (Private) 425  
adjust\_row() (Private) 425  
advance\_layer\_list() (Protected) 677  
advance\_proj() (Protected) 678  
align\_subdomain\_boundaries() (Protected) 419  
aligned\_sheet\_chords() (Public) 716  
alignment\_status() (Public) 195  
all\_hard() (Private) 575  
all\_nodes() (Public) 466  
allocate\_copied() (Private) 315  
allocate\_hex\_knowing\_nodes() (Private) 206  
allocate\_laplace() (Public) 345  
allocate\_more\_tri() (Private) 231  
allocate\_node() (Public) 759  
allocate\_node\_hex() (Public) 730  
allocate\_p1() (Private) 231  
allocate\_projection() (Protected) 678  
allocate\_tipton() (Private) 579  
allocate\_tipton() (Public) 345  
allocate\_tri() (Private) 231

# Function Index

allocate\_u\_v\_space() (Public, static) 577  
allocate\_wedge() (Public) 731  
allocate\_weight() (Public) 345  
allowed\_to\_seam() (Private) 425  
angle() (Private) 79  
angle\_at() (Protected) 580  
angle\_between() (Public) 99, 164  
angle\_type() (Public) 570  
angle\_with() (Public) 140, 439, 724  
animate\_new\_hex() (Public) 333  
animation\_in\_stream() (Private) 329  
animation\_out\_stream() (Private) 329  
another\_three\_neighbor() (Private) 108  
append() (Public) 622, 639  
append\_copied\_data() (Public) 475  
append\_faces() (Public) 83  
append\_item() (Protected) 245  
append\_link() (Protected) 200, 512  
append\_point() (Public) 520  
append\_queue() (Public) 458  
append\_to\_connected\_list() (Public) 639  
append\_unique() (Public) 200, 511  
apply\_solution() (Public) 375  
apply\_to\_edge() (Public) 94  
arc\_length() (Public) 468  
are\_next\_to() (Public) 744  
area() (Public) 59, 141, 311, 491  
Area2() (Private, static) 283  
ArrayBasedContainer (Public, constructor) 63  
ArrayBasedContainer (Public, destructor) 63  
assembly\_number() (Public) 254  
assign() (Public) 415, 541  
assign\_chords() (Public) 729  
assign\_entity\_name() (Public) 472  
assign\_intervals() (Public) 577  
assign\_material\_block() (Private, static)  
    283  
assign\_node\_ownership() (Private) 265  
assign\_nodes() (Private) 265  
assign\_ownership() (Public) 345  
assign\_sizing\_function() (Public) 267  
assign\_target\_face\_ownership() (Protect-  
    ed) 676  
assign\_void\_ownership() (Private) 331  
associate() (Public) 351  
associate\_element() (Private) 266  
associate\_nodes() (Private) 265  
associate\_to\_LP() (Public) 374  
at() (Public) 219, 236, 237  
attach\_nearest\_node() (Public) 83  
attached\_nodes() (Public) 566, 638  
attached\_nodes\_new() (Public) 638  
ATTRIB\_CUBIT\_OWNER (Public, construc-  
    tor) 67  
ATTRIB\_GTC\_NAME (Public, constructor)  
    71  
ATTRIB\_PARENTS (Public, constructor) 73  
attribute\_value() (Public) 257, 259  
attribute\_value() (Public, virtual) 254, 297  
auto\_compute() (Public) 94  
auto\_set\_vertex\_types() (Public) 79  
AutoVertexType (Public, constructor) 79  
AutoVertexType (Public, destructor) 79  
average\_angle\_neighbor() (Public) 440  
average\_angle\_with() (Public) 440  
average\_edge\_length() (Public) 706  
average\_location() (Public) 165  
average\_nodes\_per\_occupied\_cell()  
    (Public) 324  
average\_thickness() (Protected) 677

---

**B**

back() (Public) 199  
background\_color() (Public) 221, 238  
barset\_handler() (Public, static) 283  
base\_append() (Protected) 548  
base\_chord() (Public) 730  
base\_face() (Public) 148  
base\_faces() (Private) 445  
base\_insert() (Protected) 548  
base\_next() (Public) 549  
base\_pop() (Protected) 181  
base\_push() (Protected) 181  
base\_ref\_face() (Public) 438  
base\_remove() (Protected) 548  
bc\_handler() (Public, static) 283  
beg\_face\_id() (Public) 716  
begin\_facet\_list() (Private) 231  
beginning\_face() (Public) 714  
best\_another\_intersect() (Private) 426  
best\_repeated\_sheetchord() (Public) 759

# Function Index

best\_self\_intersect() (Private) 426  
binary\_search() (Private) 512  
bl\_attribute\_handler() (Public, static) 77  
blade\_edge() (Public) 557  
blade\_faces() (Public) 148  
BladeEdges (Public, constructor) 89  
blind\_chord\_list() (Public) 742  
blind\_incomplete\_chords() (Public) 731  
block\_id() (Public) 473  
block\_list() (Public) 299  
block\_size() (Public) 381  
blow\_handler() (Public, static) 389  
blow\_out() (Public) 742  
body\_handler() (Public, static) 283  
body\_transformations() (Public, static) 303  
BODYs\_interfering() (Public) 57  
boolean\_handler() (Public, static) 303  
border\_v() (Private) 110  
both\_nodes() (Public) 710  
both\_whiskers() (Public) 724  
boundary\_faces() (Public) 503  
boundary\_layer() (Public) 96  
boundary\_layer\_edges() (Public) 99  
boundary\_layer\_faces() (Public) 397  
boundary\_layer\_growth() (Public) 96  
boundary\_layer\_id() (Public) 96  
boundary\_layer\_number\_elements() (Public) 96  
boundary\_layer\_on\_edge() (Public) 99  
boundary\_layers() (Public) 397  
boundary\_smooth() (Protected) 580  
boundary\_transition() (Private) 101  
BoundaryLayer (Public, constructor) 93  
BoundaryLayer (Public, destructor) 93  
BoundaryLayerEdge (Public, constructor) 95  
BoundaryLayerEdge (Public, destructor) 95  
BoundaryLayerFace (Public, constructor) 99  
BoundaryLayerFace (Public, destructor) 99  
BoundaryLayerTool (Public, constructor) 102  
BoundaryLayerTool (Public, destructor) 102  
bounding\_box() (Public) 59, 496  
bounding\_box() (Public, virtual) 473  
bounding\_cell\_maximum\_x() (Public) 324  
bounding\_cell\_maximum\_y() (Public) 324  
bounding\_cell\_maximum\_z() (Public) 324  
bounding\_cell\_minimum\_x() (Public) 324

bounding\_cell\_minimum\_y() (Public) 324  
bounding\_cell\_minimum\_z() (Public) 324  
bounding\_range() (Private) 324  
bounding\_range\_maximum() (Public) 324  
bounding\_range\_minimum() (Public) 324  
break\_chord() (Private, static) 754  
break\_loop() (Public) 745  
break\_neighbors() (Public) 745  
brick() (Public) 310  
brick\_volume() (Public) 552  
broken() (Private) 739  
broken() (Public) 745  
build\_active\_node\_list() (Protected) 676  
build\_check\_edges() (Private) 428  
build\_component() (Private) 433  
build\_corner\_states() (Public) 756  
build\_pyramid() (Private) 445  
build\_shrink\_set() (Private) 207  
build\_tree() (Private) 79

## C

---

c\_str() (Public) 184  
calc\_translation\_vector() (Private) 666  
calculate() (Public) 453  
calculate\_2d\_quad\_quality() (Private) 447  
calculate\_chord\_angle() (Public) 741  
calculate\_element\_quality() (Public) 341, 447  
calculate\_element\_quality() (Public, virtual) 263  
calculate\_extended\_point() (Public) 520, 742  
calculate\_extended\_point() (Public, static) 520  
calculate\_length() (Public) 468  
calculate\_new\_chord\_angle() (Public) 745  
calculate\_rotation\_parameters() (Private) 510  
calculate\_sheetschord\_angle() (Public) 741  
camera\_move() (Public) 220, 238  
canCollapse() (Public) 736  
canMerge() (Private) 735  
canOpenSurfaceCrack() (Public, static)

# Function Index

149

capture\_quality() (Private) 301  
cast\_to\_BLE() (Public, virtual) 474  
cast\_to\_cell\_index() (Public) 597  
cast\_to\_cell\_index() (Public, virtual) 659  
cast\_to\_center\_node() (Public) 599  
cast\_to\_center\_node() (Public, virtual) 659  
cast\_to\_copied() (Public) 601  
cast\_to\_copied() (Public, virtual) 659  
cast\_to\_cubit\_face() (Public) 141  
cast\_to\_cubit\_face() (Public, virtual) 387  
cast\_to\_Distance() (Public) 603  
cast\_to\_distance() (Public, virtual) 659  
cast\_to\_exodus\_mesh() (Public) 268  
cast\_to\_exodus\_mesh() (Public, virtual) 474  
cast\_to\_full\_hex() (Public) 290  
cast\_to\_full\_hex() (Public, virtual) 144  
cast\_to\_genesis\_id() (Public) 605  
cast\_to\_genesis\_id() (Public, virtual) 659  
cast\_to\_geometry\_sizing() (Public) 607  
cast\_to\_geometry\_sizing() (Public, virtual) 659  
cast\_to\_hex\_knowing() (Public) 609  
cast\_to\_hex\_knowing() (Public, virtual) 659  
cast\_to\_laplace() (Public) 611  
cast\_to\_laplace() (Public, virtual) 659  
cast\_to\_layer() (Public) 613  
cast\_to\_layer() (Public, virtual) 659  
cast\_to\_line() (Public) 278  
cast\_to\_line() (Public, virtual) 277  
cast\_to\_lp\_edge() (Public) 615  
cast\_to\_lp\_edge() (Public, virtual) 659  
cast\_to\_mapping\_face() (Public) 374  
cast\_to\_mapping\_face() (Public, virtual) 659  
cast\_to\_morph() (Public) 619  
cast\_to\_morph() (Public, virtual) 659  
cast\_to\_node\_hard\_line() (Public) 621  
cast\_to\_node\_hard\_line() (Public, virtual) 659  
cast\_to\_node\_hex() (Public) 412  
cast\_to\_node\_hex() (Public, virtual) 144  
cast\_to\_normal() (Public) 623  
cast\_to\_normal() (Public, virtual) 659  
cast\_to\_paver() (Public) 625  
cast\_to\_paver() (Public, virtual) 659

cast\_to\_pillow() (Public) 434, 627  
cast\_to\_pillow() (Public, virtual) 659, 741  
cast\_to\_point() (Public) 278  
cast\_to\_point() (Public, virtual) 277  
cast\_to\_projection() (Public) 631  
cast\_to\_projection() (Public, virtual) 659  
cast\_to\_proto\_hex() (Public) 438  
cast\_to\_proto\_hex() (Public, virtual) 273  
cast\_to\_ref\_body() (Public, virtual) 463, 474  
cast\_to\_ref\_edge() (Public, virtual) 469, 474  
cast\_to\_ref\_face() (Public, virtual) 474, 490  
cast\_to\_ref\_group() (Public, virtual) 474, 495  
cast\_to\_ref\_vertex() (Public, virtual) 474, 499  
cast\_to\_ref\_volume() (Public, virtual) 474, 505  
cast\_to\_region() (Public) 279  
cast\_to\_region() (Public, virtual) 277  
cast\_to\_sheet\_chord() (Public) 522  
cast\_to\_sheet\_chord() (Public, virtual) 387  
cast\_to\_side() (Public) 280  
cast\_to\_side() (Public, virtual) 277  
cast\_to\_sizing() (Public) 633  
cast\_to\_sizing() (Public, virtual) 659  
cast\_to\_strider() (Public) 635  
cast\_to\_strider() (Public, virtual) 659  
cast\_to\_sub\_map() (Public) 565, 638  
cast\_to\_sub\_map() (Public, virtual) 659  
cast\_to\_tipton() (Public) 643  
cast\_to\_tipton() (Public, virtual) 659  
cast\_to\_u\_v\_space() (Public) 647  
cast\_to\_u\_v\_space() (Public, virtual) 659  
cast\_to\_vol\_map() (Private) 649  
cast\_to\_weight() (Public) 651  
cast\_to\_weight() (Public, virtual) 659  
cast\_to\_whisker\_face() (Public) 723  
cast\_to\_whisker\_face() (Public, virtual) 659  
cast\_to\_whisker\_hex() (Public) 731  
cast\_to\_whisker\_hex() (Public, virtual) 387, 659  
cell\_A() (Public) 90

# Function Index

cell\_B() (Public) 90  
cell\_begin\_A() (Public) 89  
cell\_begin\_B() (Public) 89, 90  
cell\_end\_A() (Public) 89, 90  
cell\_end\_B() (Public) 89, 90  
cell\_from\_range() (Private) 324  
cell\_index() (Public) 597  
center() (Public) 125, 140, 144, 220, 238, 439  
center() (Public, static) 140  
center\_node() (Public) 133, 141, 290, 412, 599  
center\_node() (Public, virtual) 144  
center\_point() (Public) 462, 468, 485, 498, 504  
center\_point() (Public, virtual) 473  
centroid\_area\_pull\_smooth() (Private) 579  
centroid\_area\_pull\_smooth\_controller() (Private) 579  
change\_cell() (Public) 325  
change\_chords\_owning\_sheet() (Private) 740  
change\_dir() (Public) 639  
change\_face\_sheet() (Public) 724  
change\_item\_to() (Protected) 201, 512  
change\_node() (Public) 134  
change\_proto\_hex\_state() (Public) 83  
change\_toContaining\_faces() (Public, static) 474  
change\_toFullHex() (Public, static) 144  
check\_adjacent\_chords() (Public) 758  
check\_and\_set\_reading\_fastq() (Private, static) 284  
check\_angle() (Public) 640  
check\_axis\_vector\_direction() (Private) 510  
check\_bodies() (Public) 60  
check\_boundaries() (Private) 585  
check\_bounding\_box() (Private) 426  
check\_chords() (Public) 744  
check\_curves() (Public) 60  
check\_equal\_radius() (Private, static) 283  
check\_face\_intersections() (Private) 551  
check\_for\_closed\_loop() (Public) 521  
check\_for\_doublets() (Public) 747  
check\_for\_hexes() (Private) 579  
check\_hard\_lines() (Public) 581  
check\_hex\_cells() (Public) 759  
check\_intersections() (Private) 428  
check\_intervals() (Public) 571, 577, 671  
check\_inversion() (Public) 730  
check\_join() (Public) 747  
check\_joins() (Public) 757  
check\_loops() (Private) 585  
check\_merge() (Public) 746, 758  
check\_merge\_2() (Public) 746  
check\_merges() (Public) 757  
check\_normals() (Private) 585  
check\_opposite\_intervals() (Private) 576  
check\_pillow() (Private) 433  
check\_point() (Public) 528  
check\_region\_order() (Private, static) 284  
check\_sheet\_merge() (Public) 715  
check\_sheets() (Public) 729  
check\_side\_mapping\_intervals() (Public) 374  
check\_sizing\_data() (Public) 490  
check\_state() (Public) 757  
check\_surfaces() (Public) 60  
check\_third() (Public) 746  
check\_third\_chord() (Public) 744  
check\_to\_squeeze() (Private) 112  
check\_unfinished\_states() (Public) 760  
check\_valid() (Public) 517  
check\_validity() (Private) 693  
check\_values() (Private) 266  
check\_weave\_state() (Public) 521  
check\_whiskers() (Public) 759  
chevron\_face() (Private) 111  
chord() (Public) 723  
chord\_1\_order() (Public) 730  
chord\_2\_order() (Public) 730  
chord\_3\_order() (Public) 730  
chord\_angle() (Public) 741  
chord\_color() (Private) 232  
chord\_point\_list() (Public) 741  
chord\_points() (Public) 709  
class\_name() (Public, virtual) 137  
classify\_boundary\_nodes() (Protected) 678  
classify\_corner() (Private) 694

# Function Index

classify\_end() (Private) 694  
classify\_face() (Private) 694  
classify\_node\_face() (Private) 695  
classify\_ref\_edges() (Private) 695  
classify\_side() (Private) 694  
classify\_six() (Private) 694  
classify\_virt\_edge() (Public) 571  
classify\_virtual\_face() (Public, virtual)  
    697  
classify\_volume() (Private) 694  
classify\_with\_out() (Private) 695  
clean\_mesh\_on\_boundary() (Private) 109  
clean\_out() (Private, static) 283  
clean\_out() (Public) 63, 200, 397, 714  
clean\_out\_edges() (Public) 625  
clean\_out\_lists() (Public) 639  
clean\_type() (Public) 141  
clean\_up() (Public) 141  
clean\_up\_connectivity() (Private) 105  
clean\_up\_mesh() (Public) 113  
clean\_up\_projections() (Protected) 678  
clean\_up\_topology() (Private) 105  
cleanout\_deactivated\_geometry() (Public)  
    398  
cleanout\_temporary\_geometry() (Public)  
    398  
CleanUp (Public, constructor) 112  
CleanUp (Public, destructor) 113  
cleanup\_exodus() (Public) 269  
clear() (Public) 220, 234  
clear\_chord\_segment() (Public) 220, 234  
clear\_echo\_stream() (Protected, static) 116  
clear\_neighbor\_marks() (Public) 525  
clear\_non\_retained() (Public) 220, 234  
clear\_retained() (Private) 231  
clear\_sheet\_segment() (Public) 220, 234  
close\_2() (Private) 424  
close\_4() (Private) 424  
close\_6() (Public) 581  
close\_circle() (Public) 581  
close\_intersecting\_faces() (Public) 745,  
    760  
close\_journal\_stream() (Public) 681  
close\_loop() (Private) 424  
close\_one\_center() (Public) 581  
close\_recording\_stream() (Public) 681  
close\_two\_center() (Public) 581  
close\_void() (Public) 334  
closed() (Public) 462, 502  
closed\_void() (Private) 703  
closedBody() (Public) 462  
closest\_exterior\_node() (Private) 570  
closest\_point() (Public) 60, 314  
coedge\_reversed() (Public) 96  
coefficient() (Public) 171  
cofactor() (Public) 153  
collapse() (Public) 149, 736  
collapse\_face() (Public) 165  
collapse\_face() (Public, static) 760  
collapse\_steps() (Public) 737  
collapse\_wedge() (Public) 746  
collapsed\_face() (Private) 206  
collect\_faces\_of\_physical\_sheet()  
    (Public) 747  
collect\_intervals() (Private) 687  
color() (Public) 297, 469, 485, 502  
color() (Public, virtual) 137, 472, 741  
color\_handler() (Public, static) 321  
color\_index() (Private) 232  
combine\_on\_other\_side() (Private) 111  
combine\_on\_this\_side() (Private) 111  
combine\_with\_neighbor() (Private) 111  
command\_is\_echoed() (Public) 682  
command\_is\_journaled() (Public) 682  
command\_is\_recorded() (Public) 682  
command\_wrapup() (Protected, static) 115  
comment\_handler() (Public, static) 403  
common\_chord() (Public) 730  
common\_draw\_label() (Public) 474  
common\_hex() (Public) 140, 710  
common\_ref\_edge() (Public) 490  
common\_sheet() (Public) 715  
commonRefVertex() (Public) 467  
compare() (Public) 56  
compare\_alignment() (Public) 56  
compare\_refedges() (Public) 314  
compatible() (Private) 257, 259, 261  
compatible() (Protected, virtual) 255  
complete() (Public) 566, 638  
complete\_twin\_chords() (Private) 713  
completed() (Public) 714  
composite\_angle\_at() (Private) 427

# Function Index

compress() (Public) 200  
compress() (Public, static) 403  
compress\_all\_object\_memory() (Public, static) 384  
compress\_ids() (Public) 399  
compress\_memory() (Public) 384  
compress\_memory() (Public, static) 668  
compress\_object\_memory() (Public) 668  
compress\_object\_memory() (Public, static) 384  
compute\_alignment() (Private) 195  
compute\_growth\_factor() (Private) 93  
compute\_hex\_coordinates() (Public) 731  
compute\_metric() (Private) 576  
compute\_new\_position() (Private) 510, 666  
compute\_num\_element\_blocks() (Private) 287, 300, 765  
compute\_spacing() (Private) 93  
connect\_across() (Public) 581  
connect\_and\_remove() (Public) 639  
connect\_at\_node() (Private) 427  
connect\_copies() (Public) 637  
connect\_inner\_loop() (Private) 570  
connect\_loops() (Private) 427  
connect\_node() (Public) 637  
connect\_sub\_nodes() (Private) 696  
connect\_virtual\_surface\_nodes() (Private, virtual) 696  
connected\_through\_boundary() (Private) 426  
connects\_to() (Public) 279  
consolidate\_ref\_edges() (Public) 488  
constant\_sizing\_function() (Private) 484  
construct\_2cells() (Public) 747  
construct\_3cell() (Public) 711  
construct\_all\_3cells() (Public) 759  
construct\_face() (Private) 551  
construct\_nodes() (Public) 730  
construct\_primal() (Public) 758  
construct\_volume\_cells() (Private) 754  
containing\_ref\_entities() (Public) 469, 490, 499, 505  
containing\_ref\_entities() (Public, virtual) 462, 474, 496  
contains() (Public) 99, 134, 139, 140, 144, 164, 467, 488, 489  
contains\_chord() (Public) 731  
contains\_EDGE() (Public) 490  
contains\_entities() (Public) 710  
contains\_node\_location() (Public) 498  
contains\_ref\_edge() (Public) 94  
control\_list() (Public) 607  
ControlPoint (Public, constructor) 119  
convert\_loop() (Public) 378  
convert\_to\_cub() (Public) 378  
convert\_to\_sub() (Public) 378  
coordinate\_system() (Private) 585  
coordinates() (Public) 164, 498  
copied\_node() (Public) 601  
copied\_nodes() (Public) 622  
copiedFromId() (Public) 461  
copy() (Public) 463, 724  
copy() (Public, virtual) 473  
copy\_BODY() (Public) 56  
copy\_body() (Public) 312  
copy\_hard\_line\_nodes() (Public) 582  
copy\_list() (Public) 696  
copy\_mesh() (Public) 312  
copy\_mesh() (Public, static) 389  
copy\_node() (Public) 639, 640  
copy\_node\_list() (Private) 101  
copy\_solid\_mesh() (Public) 313  
copy\_surf\_mesh() (Public) 313  
corner\_angle\_metric() (Protected) 377  
corner\_edge() (Private) 109  
corner\_node() (Public) 640  
corner\_star() (Private) 110  
corner\_traversal() (Private) 695  
corner\_type() (Public) 566, 639  
corresponding\_points() (Public) 528  
count\_edges() (Public) 561  
count\_stream\_users() (Private) 156  
count\_triangular\_elements() (Public) 673  
cpu\_secs() (Public) 123, 653  
CpuTimer (Public, constructor) 123  
crack\_mesh() (Private) 109  
crack\_on\_left() (Private) 109  
crack\_on\_right() (Private) 109  
crack\_periodic\_surface() (Public) 577  
create() (Public) 102, 428  
create\_background\_mesh() (Public) 419  
create\_boundary() (Private) 587

# Function Index

create\_boundary\_layer\_edge\_list() (Private) 101  
create\_boundary\_layer\_edge\_lists() (Public) 99  
create\_boundary\_lists() (Public) 83  
create\_center\_node() (Public) 133, 140, 290, 412  
create\_center\_node() (Public, virtual) 144  
create\_curve\_mesh() (Private) 266  
create\_edge() (Private) 330, 347  
create\_edge() (Public, virtual) 83  
create\_edges() (Private) 561  
create\_elements\_above\_edge() (Private) 101  
create\_end\_mesh\_box() (Private) 101  
create\_face() (Public) 83  
create\_face\_uses() (Private) 289  
create\_faces() (Private) 575  
create\_full\_hex() (Public) 87  
create\_geometry() (Public, static) 303  
create\_hexes() (Private) 687  
create\_hexes() (Protected) 676  
create\_infinite\_plane\_cutting\_tool() (Public) 311  
create\_instance() (Public) 224  
create\_knife() (Public) 148  
create\_left\_node\_list() (Private) 101  
create\_mesh() (Protected, virtual) 677  
create\_mesh() (Public) 249, 332, 350, 445, 498, 509, 552, 561, 577, 585, 595, 665, 671  
create\_mesh\_box() (Private) 101  
create\_necklace() (Private) 679  
create\_neighbor\_array() (Public) 745  
create\_neighbor\_arrays() (Public) 745  
create\_new\_entity() (Public) 300  
create\_node() (Private) 330, 347  
create\_node() (Public, virtual) 83  
create\_offset\_curve() (Private) 102  
create\_parallel\_mesh() (Public) 419  
create\_pillows() (Public, static) 434  
create\_projected\_nodes() (Private) 101  
create\_projection\_memory() (Protected) 678  
create\_projection\_nodes() (Private) 510, 595, 665  
create\_projection\_nodes() (Protected, virtual) 677  
create\_projection\_nodes\_new() (Private) 595  
create\_quality\_controller() (Protected) 471  
create\_quality\_controller() (Public, static) 453  
create\_quality\_instance() (Private, virtual) 343, 449, 453  
create\_ref\_face() (Private) 587  
create\_right\_node\_list() (Private) 101  
create\_sheets() (Public) 756  
create\_solid\_geometry() (Public) 268  
create\_super\_acis\_bounding\_box() (Public) 58  
create\_whisker\_faces() (Public, static) 724  
crossing\_sheet() (Public) 724  
cubit\_edge() (Public) 709  
cubit\_entity\_list() (Public) 399  
cubit\_face() (Public) 723  
cubit\_face\_factory() (Private) 427  
cubit\_face\_list() (Public) 742  
cubit\_faces() (Public) 486  
cubit\_node\_factory() (Private) 112  
cubit\_nodes() (Public) 730  
cubit\_owner() (Public) 67  
cubit\_revision\_date() (Public, static) 682  
cubit\_version() (Public, static) 682  
CubitBox (Public, constructor) 125  
CubitBox (Public, destructor) 125  
CubitCollection (Public, constructor) 127  
CubitCollection (Public, virtual, destructor) 127  
CubitContainer (Public, constructor) 129  
CubitContainer (Public, virtual, destructor) 129  
CubitEdge (Public, constructor) 133  
CubitEdge (Public, destructor) 133  
CubitEntity (Public, constructor) 137  
CubitEntity (Public, virtual, destructor) 137  
CubitFace (Public, constructor) 139  
CubitFace (Public, destructor) 139  
CubitHex (Public, constructor) 143  
CubitHex (Public, virtual, destructor) 143  
CubitKnife (Public, constructor) 148  
CubitKnife (Public, destructor) 148

# Function Index

CubitMatrix (Public, constructor) 153  
CubitMatrix (Public, destructor) 153  
CubitMessage (Protected, constructor) 156  
CubitMessage (Public, destructor) 156  
CubitNode (Public, constructor) 162  
CubitNode (Public, destructor) 162  
CubitNodeArray (Public, constructor) 170  
CubitNodeArray (Public, destructor) 170  
CubitNodeArray1D (Public, constructor) 167  
CubitNodeArray1D (Public, destructor) 167  
CubitNodeArray2D (Public, constructor) 169  
CubitNodeArray2D (Public, destructor) 169  
CubitPlane (Public, constructor) 171  
CubitPtrArray1D (Public, constructor) 175  
CubitPtrArray1D (Public, destructor) 175  
CubitPtrArray2D (Public, constructor) 176  
CubitPtrArray2D (Public, destructor) 176  
CubitPtrArray3D (Public, constructor) 177  
CubitPtrArray3D (Public, destructor) 177  
CubitSheet (Public, constructor) 179  
CubitSheet (Public, destructor) 179  
CubitStack (Public, constructor) 181  
CubitStack (Public, virtual, destructor) 181  
CubitString (Public, constructor) 183  
CubitString (Public, destructor) 183  
CubitVoxel (Public, constructor) 191  
CubitVoxel (Public, destructor) 191  
cull\_global\_list() (Public, static) 134, 141, 144, 165  
cull\_global\_mesh\_lists() (Public) 398  
current\_allocated\_memory() (Public, static) 63  
current\_body\_id() (Public) 398  
current\_child\_item() (Public) 668  
current\_edge\_id() (Public) 398  
current\_face\_id() (Public) 398  
current\_group\_id() (Public) 398  
current\_vertex\_id() (Public) 398  
current\_volume\_id() (Public) 398  
currently\_starred() (Public) 211, 431, 627  
curvature\_sizing\_function() (Private) 484  
curve\_aligned() (Public) 588  
curve\_color() (Private) 232  
curve\_subdomains() (Public) 588  
curves\_segment() (Public) 221, 239

CurveSubDomain (Public, constructor) 195  
CurveSubDomain (Public, destructor) 195  
cut\_last\_hex() (Public) 520, 716  
cut\_link() (Protected) 200  
cylinder() (Public) 310

---

**D**

data\_valid() (Protected) 453  
de\_activate() (Public) 737  
de\_allocate\_copied() (Private) 315  
de\_allocate\_hex\_knowing\_nodes() (Private) 206  
de\_allocate\_laplace() (Public) 345  
de\_allocate\_projection() (Protected) 678  
de\_allocate\_tipton() (Private) 579  
de\_allocate\_tipton() (Public) 345  
de\_allocate\_u\_v\_space() (Public, static) 577  
de\_allocate\_weight() (Public) 345  
deactivated() (Public) 473  
deactivateRefStructure() (Public) 474  
deallocate() (Private) 231  
deallocate\_morph() (Private) 585  
debug() (Public) 215, 234, 653  
debug\_draw() (Public) 741  
debug\_flag() (Public) 156  
decompose() (Public) 312  
decompose\_surface\_mesh() (Protected) 419  
decrement\_offset() (Private) 438  
decrement\_use\_count() (Public) 473  
default\_handler() (Public, static) 283  
default\_scheme() (Public, static) 469, 491, 505  
define\_command\_options() (Private) 682  
define\_rotate\_general\_vectors() (Private, static) 321  
degree\_2cell() (Public) 747  
deletable() (Public) 134, 141, 144, 162  
delete\_ACIS\_BODY() (Public) 55  
delete\_all() (Public, static) 134, 140, 143, 164, 746  
delete\_bare\_blinds() (Public) 760  
delete\_block() (Public) 299  
delete\_body() (Private, static) 303  
delete\_boundary\_layer\_faces() (Public)

# Function Index

398  
delete\_boundary\_layers() (Public) 398  
delete\_curve\_mesh() (Private, static) 389  
delete\_edge() (Public) 83  
delete\_entity\_mesh() (Private, static) 389  
delete\_geometry() (Public) 398  
delete\_handler() (Public, static) 303  
delete\_instance() (Public) 224  
delete\_internal\_face() (Public) 87  
delete\_mapping\_faces() (Public, static) 577  
delete\_mesh() (Public) 398  
delete\_mesh() (Public, static) 389  
delete\_mesh\_entities() (Public) 398  
delete\_meshes\_of\_deactivated\_refEntit  
ies() (Private) 400  
delete\_node() (Public) 83  
delete\_node\_remove() (Public) 83  
delete\_paver\_node() (Private) 427  
delete\_picture\_segment() (Public) 235  
delete\_proto\_hex() (Public) 87  
delete\_refbodies() (Public) 309  
delete\_surface\_mesh() (Private, static) 389  
delete\_surface\_mesh() (Public) 571  
delete\_TD() (Public) 663  
delete\_this\_face() (Private) 111  
delete\_tree() (Public) 667  
delete\_unused\_edges() (Public) 164  
delete\_vertex\_mesh() (Private, static) 389  
delete\_volume\_mesh() (Private, static) 389  
delete\_whisker\_faces() (Public, static) 724  
delete\_whisker\_sheets() (Public) 760  
deleted\_flag() (Public) 714  
deleted\_limit() (Public) 755  
deleted\_list() (Public) 755  
DeltaTime (Private, constructor) 655  
DeltaTime (Public, constructor) 655  
denom() (Public) 643  
density() (Public) 607, 635  
density\_factor() (Public) 307, 561  
depth() (Public) 668  
destroy\_associativity\_nodesets() (Pub  
lic) 300  
destroy\_memory() (Public) 119, 374, 597,  
599, 601, 603, 605, 607, 609, 611, 613,  
615, 619, 621, 623, 625, 628, 631, 633,  
635, 639, 644, 647, 651  
destroy\_memory() (Public, static) 119, 374,  
384, 440, 597, 599, 601, 603, 605, 607,  
609, 611, 613, 615, 619, 621, 623, 625,  
628, 631, 633, 635, 639, 644, 647, 651  
destroy\_projection\_memory() (Protected)  
678  
detach\_refentity\_from\_parent\_ENTITY()  
(Public) 474  
detect\_breakout() (Public) 491  
determinate() (Public) 153  
determine\_cutting\_plane() (Private) 693  
determine\_location() (Public) 192, 706  
determine\_loop() (Private) 331  
determine\_six\_dir() (Private) 696  
DFS\_next\_item() (Public) 668  
DFS\_next\_leaf\_item() (Public) 668  
DFS\_start\_item() (Public) 668  
diagnostic\_flag() (Public) 156  
diagonal() (Public) 125  
diagonal\_edge() (Private) 330, 348  
diagonal\_edge() (Public, virtual) 83  
diagonal\_node() (Public) 140  
difference\_face\_list() (Public) 83  
dimension() (Public) 469, 490, 499, 505  
dimension() (Public, virtual) 474  
direction() (Public) 613  
disect\_loops() (Private) 570  
display() (Public) 216, 234  
display() (Public, static) 321  
display\_all() (Public) 222, 239  
dist\_apart() (Private) 445  
distance() (Public) 172, 439, 603, 633  
distance\_squared() (Private) 428  
distinct\_2cell\_edges() (Private) 754  
distinct\_2cell\_vertices() (Private) 754  
distinct\_edge\_vertices() (Private) 754  
distribute() (Public) 573  
DLLList (Public, constructor) 199  
DLLList (Public, destructor) 199  
do\_you\_cross() (Public) 746  
does\_instance\_exist() (Public) 224  
done\_weaving() (Public) 744  
double\_five\_left() (Private) 108  
double\_five\_right() (Private) 108  
double\_row\_wing() (Private) 110  
double\_three() (Private) 106

# Function Index

double\_triangle\_left() (Private) 110  
double\_triangle\_right() (Private) 110  
double\_twist\_left() (Private) 110  
double\_twist\_right() (Private) 110  
doublet\_cell() (Public) 710  
doublet\_hex() (Public) 731  
doublet\_opposite\_point() (Public) 527  
doublet\_points() (Public) 756  
DoubletPillower (Public, constructor) 208  
down\_grade\_owner() (Private) 207  
draw() (Public) 94, 96, 134, 148, 163, 179,  
195, 222, 240, 253, 267, 274, 290, 412,  
415, 453, 462, 467, 487, 495, 498, 504,  
519, 542, 558, 587, 701, 709, 716, 729,  
741  
draw() (Public, virtual) 137, 141, 143, 297,  
472  
draw() (Public, virtual, virtual) 387  
draw\_all() (Private) 428  
draw\_all\_hidden() (Private) 331  
draw\_all\_sheets() (Public) 757  
draw\_all\_volume\_sheets() (Public, static)  
334  
draw\_blind\_chord() (Private) 519  
draw\_boundary() (Private) 331  
draw\_boundary\_hidden() (Private) 331, 348  
draw\_boundary\_normals() (Private) 331  
draw\_by\_hex() (Public, static) 334  
draw\_cell\_range() (Public) 325  
draw\_chord() (Private) 519  
draw\_doublet() (Private) 208  
draw\_edge() (Private) 428  
draw\_edges() (Public) 141, 346, 395  
draw\_face\_retain() (Public) 334  
draw\_faces() (Public) 83  
draw\_geometry\_labels() (Protected) 233  
draw\_grid() (Public) 325  
draw\_grid\_cell() (Public) 325  
draw\_grid\_cell\_nodes() (Public) 325  
draw\_grid\_cell\_range() (Public) 325  
draw\_grid\_cell\_range\_nodes() (Public)  
325  
draw\_handler() (Private, static) 451  
draw\_handler() (Public, static) 321  
draw\_hex\_nodes() (Public) 334  
draw\_hidden\_sheet\_faces() (Public) 179  
draw\_histogram() (Public) 223, 241  
draw\_id() (Public) 334  
draw\_label() (Public) 134, 140, 143, 163,  
222, 240, 439, 462, 467, 487, 498, 504  
draw\_logical\_loop() (Public) 742  
draw\_loop() (Private) 587  
draw\_loop() (Protected) 580  
draw\_loop() (Public) 760  
draw\_loop\_faces() (Public) 742  
draw\_mesh() (Private) 112, 561, 587  
draw\_mesh() (Protected) 419, 580  
draw\_mesh() (Public) 462, 467, 487, 495, 498,  
504  
draw\_mesh() (Public, virtual) 137, 472  
draw\_mesh\_labels() (Protected) 233  
draw\_my\_edges() (Public) 487  
draw\_my\_faces() (Public) 504  
draw\_neighbor\_faces() (Public) 335  
draw\_node\_list() (Private) 112  
draw\_now() (Public) 137  
draw\_one\_sheet() (Public, static) 757  
draw\_owned\_faces() (Public) 487  
draw\_owned\_nodes() (Public) 462, 467, 487,  
504  
draw\_physical\_chord() (Public) 717  
draw\_physical\_sheet() (Public) 746, 760  
draw\_physical\_sheets() (Public) 760  
draw\_physical\_weave() (Public) 758  
draw\_proto() (Public) 439  
draw\_proto\_hex\_from\_face\_list() (Pub-  
lic) 333  
draw\_proto\_normal() (Public) 439  
draw\_proto\_on\_boundary() (Private) 348  
draw\_pyramid() (Public) 443  
draw\_sheet() (Public) 741  
draw\_sheet\_edge() (Public) 535  
draw\_sheet\_faces() (Public) 179  
draw\_sheets() (Public) 332  
draw\_shrink\_set() (Private) 208  
draw\_sides() (Private) 373  
draw\_smooth\_set() (Protected) 580  
draw\_stars() (Private) 208  
draw\_three\_hex\_sheets() (Public, static)  
334  
draw\_three\_sheets() (Public, static) 757  
draw\_valence\_check() (Private) 112

# Function Index

draw\_vertices() (Public) 462, 467, 487, 504  
draw\_voids() (Public) 504  
drawing\_mode() (Public) 219, 236  
DrawingTool (Protected, constructor) 215  
DrawingTool (Public, destructor) 215  
DrawingToolInstance (Public, constructor) 234  
DrawingToolInstance (Public, destructor) 234  
drawMyVolumes() (Public) 462  
drive() (Private) 736  
drive() (Public) 149  
drive\_edges() (Public) 148  
drive\_from\_root() (Private) 736  
drive\_node() (Public) 148  
drive\_nodes() (Public) 148  
drive\_one\_at\_a\_time() (Public) 737  
drive\_to\_blade() (Public) 737  
drive\_wedge() (Public) 746  
dual\_edge() (Public) 748  
dummy() (Private, virtual) 77, 116, 284, 285, 295, 303, 321, 389, 403  
dump\_hardcopy() (Public) 220, 238  
dump\_postscript() (Public) 220, 238  
duplicate\_edge() (Public) 164  
DynamicArray (Public, constructor) 245  
DynamicArray (Public, destructor) 245

## E

---

edge() (Public) 192, 251, 392, 613  
edge\_associated() (Public) 351  
edge\_bound() (Public) 352  
edge\_color() (Private) 232  
edge\_column() (Public) 351  
edge\_degree\_weight() (Public) 165  
edge\_degree\_weight() (Public, static) 165  
edge\_degree\_weight\_function() (Public) 366  
edge\_degree\_weight\_on\_collapse() (Public) 724  
edge\_interior\_nodes() (Public) 96  
edge\_intersects() (Public) 192  
edge\_intersects\_voxel() (Public) 706  
edge\_list() (Public) 374  
edge\_position() (Public) 119  
edge\_reversal() (Private) 102

edge\_reversal() (Public) 99  
edge\_use() (Public) 141, 274  
edge\_uses() (Public) 133  
EdgeMeshTool (Public, constructor) 249  
EdgeMeshTool (Public, virtual, destructor) 249  
edges() (Public) 140, 163, 290, 466, 486, 503, 542  
edges() (Public, virtual) 143  
edges\_cross\_another() (Private) 426  
edges\_cross\_self() (Private) 426  
edges\_done() (Public) 640  
edges\_inclusive() (Public) 268, 486, 503  
edges\_intersect() (Public) 581  
edges\_on\_loop() (Public) 486  
edges\_segment() (Public) 221, 239  
edges\_valid\_for\_join() (Private) 426  
edgeType() (Public) 467  
EdgeUse (Public, constructor) 251  
EdgeUse (Public, destructor) 251  
eight\_corner\_picker() (Private) 687  
eight\_nine\_handler() (Public, static) 283  
elapsed\_secs() (Public) 653  
element\_attributes() (Private) 257, 259  
element\_attributes() (Protected, virtual) 255  
element\_connectivity() (Private, virtual) 259  
element\_connectivity() (Protected) 254  
element\_size() (Public) 119  
element\_type() (Public) 254, 467, 485, 502  
element\_type() (Public, virtual) 297, 473  
element\_type\_name() (Public) 254  
ElementBlock (Public, constructor) 253  
ElementBlock (Public, virtual, destructor) 253  
ElementBlock1D (Public, constructor) 257  
ElementBlock1D (Public, destructor) 257  
ElementBlock2D (Public, constructor) 259  
ElementBlock2D (Public, destructor) 259  
ElementBlock3D (Public, constructor) 261  
ElementBlock3D (Public, destructor) 261  
ElementQuality (Public, constructor) 263  
ElementQuality (Public, virtual, destructor) 263  
elements() (Public) 466, 486, 495, 498, 503  
elements() (Public, virtual) 473  
empty\_boundary\_layer\_edge\_list() (Pri-

# Function Index

- vate) 102  
empty\_left\_list() (Public) 95  
empty\_right\_list() (Public) 95  
empty\_temp\_b1\_list() (Private) 101  
empty\_temp\_boundary\_layer\_list() (Private) 102  
encode\_class\_id() (Public) 299  
end() (Public) 392  
end\_corner\_node() (Public) 640  
end\_face\_id() (Public) 717  
end\_facet\_list() (Private) 231  
end\_node() (Public) 133, 251, 640  
end\_row() (Private) 424  
end\_traversal() (Private) 695  
ending\_face() (Public) 714  
endRefVertex() (Public) 467  
enroll() (Public) 319  
entity() (Public) 392  
entity\_id() (Public) 334  
entity\_name() (Public) 472  
entity\_names() (Public) 472  
entity\_partner() (Public) 268  
entity\_type() (Public) 96, 134, 141, 144, 165, 195, 254, 268, 275, 387, 416, 463, 469, 490, 495, 499, 505, 542, 587  
entity\_type() (Public, virtual, virtual) 137, 298, 472  
equal\_angle() (Private) 579  
equal\_angle\_adjustment() (Private) 579  
error\_count() (Public) 157  
estimate\_interval\_count() (Public) 307  
event\_loop() (Private) 407  
execution\_time\_and\_date() (Public) 682  
exhaustive\_pick\_4() (Private) 575  
exodus\_mesh() (Public) 473, 488  
exodus\_sizing\_function() (Private) 484  
ExodusMesh (Public, constructor) 267  
ExodusMesh (Public, destructor) 267  
expand\_metric\_name() (Public) 453  
export\_acis() (Private, static) 285  
export\_file() (Public, static) 285  
export\_fred() (Private, static) 285  
export\_genesis() (Private, static) 285  
export\_muse() (Private, static) 285  
export\_solid\_model() (Public) 309  
export\_xpatch() (Private, static) 285  
extract\_link() (Protected) 200, 512
- 
- F**
- face() (Public) 251, 274, 392  
face\_angle() (Public) 745  
face\_close\_to\_loop() (Private) 428  
face\_color() (Private) 232  
face\_exists() (Public) 83  
face\_has\_starred\_node() (Private) 207  
face\_identity() (Public) 99  
face\_in\_shrink\_set() (Private) 206  
face\_marked() (Private) 688  
face\_on\_boundary() (Private) 206  
face\_on\_surf() (Public) 413  
face\_on\_surface() (Public) 724  
face\_opposite\_nodes() (Public) 412  
face\_over\_loop() (Private) 428  
face\_point1() (Public) 725  
face\_point2() (Public) 725  
face\_sheet\_1() (Public) 723  
face\_sheet\_2() (Public) 723  
face\_use() (Public) 290, 412  
face\_use() (Public, virtual) 143  
face\_uses() (Public) 133, 140  
faceCount() (Public) 489  
faces() (Public) 133, 148, 163, 268, 503, 542  
faces\_inclusive() (Public) 503  
faces\_on\_node() (Public) 163  
faces\_on\_open\_side() (Private, static) 148  
FaceUse (Public, constructor) 273  
FaceUse (Public, virtual, destructor) 274  
factor\_interval\_handler() (Public, static) 283  
FastqBC (Public, constructor) 279  
FastqBC (Public, destructor) 279  
FastqEntity (Public, constructor) 277  
FastqEntity (Public, virtual, virtual, destructor) 277  
FastqLine (Public, constructor) 278  
FastqModel (Protected, constructor) 277  
FastqModel (Public, destructor) 277  
FastqPoint (Public, constructor) 278  
FastqRegion (Public, constructor) 279  
FastqRegion (Public, destructor) 279  
FastqSide (Public, constructor) 280

# Function Index

FastqSide (Public, destructor) 280  
feasible\_pick\_4() (Private) 576  
file\_recording\_handler() (Public, static)  
    403  
fill\_boundary\_list() (Private) 112  
fill\_final\_layer() (Protected) 677  
fill\_four() (Private) 111  
fill\_narrow\_surface() (Private) 424  
fill\_neighbor\_gap() (Private) 434, 740  
fill\_two() (Private) 111  
fill\_Y() (Private) 112  
fill\_Y\_left() (Private) 111  
fill\_Y\_right() (Private) 111  
finalize\_target\_face() (Protected) 677  
find() (Public) 184  
find\_1\_projection\_node() (Private) 679  
find\_2\_projection\_nodes() (Private) 679  
find\_2d\_corners() (Private) 696  
find\_adjacent\_face() (Public) 165  
find\_all\_possible\_drives() (Private) 736  
find\_and\_join\_hooks() (Private) 371  
find\_and\_join\_pierced\_hooks() (Private)  
    371  
find\_and\_report\_drive\_tree() (Public)  
    737  
find\_and\_report\_superdrive() (Public)  
    737  
find\_best\_cross\_self\_intersection()  
    (Private) 427  
find\_best\_leaf() (Private) 735  
find\_b1\_edge\_node() (Private) 102  
find\_b1\_edge\_node() (Public) 249  
find\_blind\_sheetchord() (Public) 747  
find\_boundary\_edges() (Public) 711  
find\_center\_coords() (Private, static) 283  
find\_char\_parameter() (Protected, static)  
    116  
find\_chord() (Public) 729  
find\_close\_edges() (Private) 428  
find\_closest\_loop() (Private) 570, 585  
find\_closest\_nodes() (Public) 571  
find\_closest\_point() (Private) 307  
find\_COEDGE() (Public) 59  
find\_collapse\_face() (Public) 165, 439  
find\_completed\_sheet\_chord() (Public)  
    744  
find\_consecutive() (Protected) 517  
find\_corner\_node\_ptr() (Private) 102  
find\_corners() (Private) 693  
find\_correct\_outer() (Private) 570  
find\_crack\_angle() (Private) 373  
find\_crossing() (Protected) 517  
find\_cut\_direction() (Private) 695  
find\_cut\_loop() (Private) 693  
find\_cut\_node() (Private) 569  
find\_displacement\_index() (Private) 266  
find\_doublets() (Private) 207  
find\_doublets\_full() (Private) 207  
find\_doublets\_node() (Private) 207  
find\_edge() (Public, static) 134  
find\_edge\_and\_add() (Public) 706  
find\_face() (Private) 266, 273  
find\_face() (Public, static) 140  
find\_faces() (Private) 736  
find\_faces\_decomposed() (Private) 595  
find\_faces\_from\_node\_list() (Public) 333  
find\_file\_use() (Private) 156  
find\_first\_not\_of() (Public) 184  
find\_first\_of() (Public) 184  
find\_hex() (Public, static) 143  
find\_hex\_type() (Private) 206  
find\_hooks() (Private) 371  
find\_id() (Private, static) 283  
find\_int\_parameter() (Protected, static)  
    116  
find\_int\_range() (Protected, static) 115  
find\_item() (Protected) 245  
find\_keyword() (Protected, static) 116  
find\_max\_hex\_id() (Public) 399  
find\_max\_whisker\_sheet\_id() (Public) 399  
find\_merged\_nodes() (Public) 84  
find\_mid\_side\_points() (Private) 671  
find\_neighbor() (Public) 527  
find\_neighbors() (Private) 438  
find\_node() (Private) 736  
find\_node() (Public) 706  
find\_node() (Public, static) 164  
find\_nodes() (Private) 695  
find\_or\_create\_set() (Private, static) 284  
find\_orbit() (Private) 365  
find\_owner() (Private) 266  
find\_owner\_of() (Public) 397, 398

# Function Index

find\_owning\_EDGES() (Private) 315  
find\_owning\_refedges() (Public) 314  
find\_plane() (Private) 696  
find\_point\_from() (Private) 102  
find\_possible\_drives() (Private) 735  
find\_projection\_node() (Protected) 676  
find\_projection\_nodes() (Protected) 676  
find\_prop\_name\_index() (Private) 266  
find\_proto\_hex\_by\_id() (Public) 334  
find\_proto\_neighbor\_nodes() (Public) 333  
find\_real\_parameter() (Protected, static) 116  
find\_refedges() (Private) 585  
find\_refverts() (Private) 585  
find\_rep\_entity() (Public) 521  
find\_sheet\_chord() (Public) 742, 744  
find\_source\_info() (Private) 585  
find\_source\_target\_entities() (Private, static) 77  
find\_special() (Private) 208  
find\_state() (Public) 438  
find\_superdrive\_path() (Private) 736  
find\_surface\_doublets() (Private) 207  
find\_tangent() (Public) 60, 314  
find\_target\_info() (Private) 585  
find\_terminal\_edge() (Private) 434, 740  
find\_third\_chord() (Public) 744  
find\_time\_step() (Private) 266  
find\_top\_neighbor() (Private) 438  
find\_variable\_index() (Private) 266  
find\_wedge() (Public) 746  
find\_whisker\_sheet() (Public) 745  
fire\_ray() (Public) 314  
first() (Public) 458  
first\_hex() (Public) 715  
first\_layer\_depth() (Public) 94  
first\_neighbor() (Public) 526  
first\_point() (Public) 520  
firstlayer\_projection\_nodes() (Private) 595  
five\_neighbor\_improvement() (Private) 106  
fix\_around\_hole() (Private) 424  
fix\_corners() (Private) 424  
fix\_hard\_line\_copies() (Public) 582  
fix\_internal\_invalid() (Private) 111  
fix\_inversion() (Public) 412  
fix\_mights() (Public) 639  
fix\_narrow\_strip() (Private) 111  
fix\_neighbor\_orientations() (Private) 433  
fix\_neighbor\_orientations() (Public) 434  
fix\_node\_hex\_orientations() (Private) 754  
fix\_node\_hex\_orientations() (Public) 717  
fix\_node\_loops() (Private) 696  
fix\_pave\_over() (Private) 428  
fix\_traversal() (Private) 696  
fix\_Y() (Private) 109  
flush() (Public) 222, 239  
flush\_facet\_list() (Private) 231  
force\_pillow\_node() (Public) 211  
force\_sizing() (static) 267  
force\_weaver() (Public) 503  
four\_34543() (Private) 107  
four\_double\_three() (Private) 107  
four\_neighbor\_improvement() (Private) 106  
fraction\_empty\_cells() (Public) 324  
FREDTool (Protected, constructor) 287  
FREDTool (Public, destructor) 287  
freeze\_size() (Public) 352  
freeze\_size() (Public, virtual) 357  
from() (Public) 219, 236, 237  
from\_abs() (Private) 233  
from\_abs() (Public) 219, 236  
full\_hex() (Public) 728  
FullHex (Public, constructor) 289  
FullHex (Public, destructor) 289

---

## G

gather\_data() (Public) 287, 299, 765  
gather\_hexes() (Public) 760  
gather\_referenced\_nodes() (Private) 300  
gather\_whisker\_chords() (Public) 760  
generate\_default\_names() (Public) 479  
generate\_journal\_entry() (Private, static) 116  
genesis\_entity\_handler() (Public, static) 295  
genesis\_entity\_list() (Public) 398

# Function Index

genesis\_geometry\_assignment() (Private, static) 295  
genesis\_id() (Public) 387  
GenesisEntity (Public, constructor) 297  
GenesisEntity (Public, virtual, destructor) 297  
GenesisTool (Protected, constructor) 299  
GenesisTool (Public, destructor) 299  
geom\_attribute\_handler() (Public, static) 77  
geometric\_angle() (Public) 60, 314, 375  
geometricFactor() (Public) 466  
geometricFactor() (Public, virtual) 472  
geometry\_check() (Public, static) 303  
geometry\_debug() (Public, static) 303  
geometry\_entity\_list() (Public) 398  
geometry\_is\_merge\_compatible() (Public, static) 759  
GeometrySizingTool (Public, constructor) 307  
GeometrySizingTool (Public, destructor) 307  
GeometryTool (Protected, constructor) 309  
GeometryTool (Public, destructor) 309  
get() (Private) 693  
get() (Public) 153  
get\_acis\_entity\_bounding\_box() (Public) 58  
get\_active\_instance() (Public) 224  
get\_adj\_edge() (Private) 694  
get\_adjacent\_2cells() (Public) 557  
get\_allocated\_objects() (Public) 384  
get\_and\_step() (Public) 701  
get\_angle() (Public) 593  
get\_arc\_nodes() (Private) 101  
get\_at() (Public) 183  
get\_attrib\_gtc\_name() (Public) 59  
get\_attrib\_name\_and\_option() (Public) 59  
get\_attribute\_value() (Public) 257, 259  
get\_attribute\_value() (Public, virtual) 254  
get\_average\_edge\_length() (Private) 407  
get\_background\_mesh\_size() (Public) 420  
get\_blade\_face\_nodes() (Public) 412  
get\_block() (Public) 381  
get\_BODY\_of\_ENTITY() (Public) 60  
get\_boundary\_layer() (Public) 397  
get\_boundary\_layer\_face() (Public) 397  
get\_boundary\_nodes() (Protected) 677  
get\_boundary\_nodes() (Public) 346  
get\_bounding\_faces() (Private) 551  
get\_cell\_from\_points() (Public) 759  
get\_child\_entities() (Public) 469, 490, 505  
get\_child\_entities() (Public, virtual) 462, 474, 496  
get\_connected\_cells() (Public) 730  
get\_copied() (Private) 315  
get\_copied\_data\_list() (Public) 475  
get\_corresponding\_points() (Public) 522  
get\_cubit\_entity\_list() (Private, static) 321  
get\_cubit\_owner\_attrib() (Public) 60  
get\_dir\_node() (Public) 639  
get\_display() (Public) 221, 234  
get\_distance() (Protected, static) 580  
get\_distance() (static) 267  
get\_doublet\_points() (Public) 755  
get\_edge\_nodes() (Private) 101  
get\_edge\_type() (Public) 505  
get\_edges() (Public) 711  
get\_edges\_from\_nodes() (Private) 332  
get\_ENTITY\_of\_refentity() (Public) 60  
get\_entry() (Public) 358  
get\_even() (Public) 466  
get\_face\_nodes() (Private) 411  
get\_face\_nodes() (Public) 412  
get\_FACES() (Public) 56  
get\_free\_objects() (Public) 384  
get\_genesis\_entity() (Public) 397  
get\_hard\_line() (Private) 427  
get\_input() (Public) 681  
get\_intervals() (Protected) 677  
get\_intervals\_new() (Protected) 677  
get\_item() (Protected) 201  
get\_item\_and\_back() (Protected) 201  
get\_item\_and\_step() (Protected) 201  
get\_laplace() (Public, static) 346  
get\_list\_items() (Public) 746  
get\_lp\_edge() (Private) 351  
get\_mapping\_face() (Public, static) 577  
get\_matching\_edge() (Private) 678  
get\_max\_size\_of\_acis\_box() (Public) 58  
get\_memory\_allocation() (Public) 381  
get\_memory\_allocation\_increment() (Public) 384

# Function Index

get\_min\_max() (Public) 237  
get\_morph() (Private) 585  
get\_name() (Public, static) 71  
get\_need\_to\_update() (Private) 233  
get\_neighborhood\_edges() (Public) 325  
get\_neighborhood\_faces() (Public) 325  
get\_neighborhood\_hexes() (Public) 325  
get\_neighborhood\_nodes() (Public) 325  
get\_neighborhood\_nodes\_sorted() (Public) 325  
get\_next\_corner() (Private) 671  
get\_next\_file() (Private) 682  
get\_next\_loop\_node() (Private) 696  
get\_next\_seedface() (Private) 703  
get\_node() (Public) 335  
get\_node\_index() (Private) 411  
get\_nodes\_face() (Public) 412  
get\_nodesets\_and\_properties() (Private) 265  
get\_normal() (Private) 445  
get\_object\_size() (Public) 384  
get\_objective() (Public) 358  
get\_opp() (Public) 639  
get\_opp\_edge() (Private) 694  
get\_option() (Public, static) 71  
get\_other\_2cell() (Public) 730  
get\_paver() (Private) 427  
get\_periodic\_sweep\_intervals() (Protected) 676  
get\_perturbation\_proto\_hex() (Private) 331  
get\_position() (Public) 55  
get\_proj() (Private) 172  
get\_proj() (Protected) 678  
get\_proto\_hex\_face\_by\_id() (Public) 335  
get\_ref\_body() (Public) 397  
get\_ref\_body\_list() (Private, static) 304  
get\_ref\_edge() (Public) 397  
get\_ref\_edges() (Private) 545  
get\_ref\_entity() (Public) 397  
get\_ref\_entity\_list() (Private, static) 77, 451  
get\_ref\_entity\_list\_kludge() (Private, static) 77  
get\_ref\_face() (Public) 397  
get\_ref\_group() (Public) 397  
get\_ref\_vertex() (Public) 397  
get\_ref\_volume() (Public) 397  
get\_refbody\_of\_ENTITY() (Private) 315  
get\_refentity() (Public) 479  
get\_refentity\_id() (Public) 479  
get\_refentity\_name() (Public) 479  
get\_refentity\_type() (Public) 479  
get\_shared\_edge() (Public) 709  
get\_sheet() (Public) 398  
get\_sheet\_chord() (Public) 527, 729, 756  
get\_sizing() (Protected, static) 580  
get\_sizing() (static) 267  
get\_sub\_entities() (Public, virtual) 474  
get\_sub\_map() (Public, static) 378, 565, 638  
get\_surf\_vertex\_angle() (Public) 489  
get\_surf\_vertex\_list() (Public) 489  
get\_surface\_smooth\_scheme() (Private, static) 77  
get\_TD() (Private) 307  
get\_TD() (Public) 663  
get\_three\_faces() (Private) 695  
get\_three\_points() (Public) 731  
get\_tipton() (Private) 345, 579  
get\_triangle\_list() (Public) 741  
get\_type() (Public) 593, 699  
get\_u\_v() (Public) 491  
get\_u\_v() (Public, static) 577  
get\_under\_edge() (Private) 694  
get\_used\_objects() (Public) 384  
get\_valence() (Private) 109  
get\_vertex\_type() (Public) 489  
get\_void\_boundary() (Public) 87  
get\_vol\_edge\_list() (Public) 505  
get\_volume\_smooth\_scheme() (Private, static) 77  
get\_whiskers() (Public) 747  
get\_win\_id() (Public) 221, 234  
GetLongOpt (Public, constructor) 319  
GetLongOpt (Public, destructor) 319  
global\_edge\_list() (Public) 399  
global\_face\_list() (Public) 399  
global\_hex\_list() (Public) 399  
global\_node\_list() (Public) 399  
global\_smooth() (Public) 580  
global\_visibility() (Public, static) 321  
global\_whisker\_hex\_list() (Public) 399

# Function Index

global\_whisker\_sheet\_list() (Public) 399  
graphics\_attribute\_handler() (Public, static) 321  
graphics\_version() (Public) 215, 234  
greedy\_reduce() (Public) 366  
grid\_cell\_index() (Private) 324  
grid\_cell\_width() (Public) 324  
grid\_range\_maximum() (Public) 324  
grid\_range\_minimum() (Public) 324  
grid\_search\_change\_cell() (Public) 332  
grid\_search\_remove\_node() (Public) 332  
grid\_search\_validity\_check() (Public) 325  
GridSearch (Public, constructor) 324  
GridSearch (Public, destructor) 324  
growth\_factor() (Public) 94

## H

---

handle\_default\_scheme() (Private, static) 77  
handle\_group() (Private, static) 77  
handle\_sweep\_schemes() (Private, static) 77  
hard\_intervals() (Private) 369  
hard\_line() (Public) 134  
hard\_line\_loop() (Public) 582  
hard\_line\_tip() (Public) 622  
hard\_points() (Public) 489  
hardcopy() (Public, static) 403  
has\_already\_been\_drawn() (Public) 527  
has\_boundary\_layer() (Public) 96, 485  
has\_higher\_order\_elements() (Public) 268  
has\_knife() (Public) 522  
has\_mesh\_elements() (Public) 488, 504  
has\_mesh\_elements() (Public, virtual) 473  
have\_to\_merge() (Private) 445  
help() (Public, static) 115  
help\_handler() (Public, static) 115  
heuristically\_improve() (Private) 576  
hex() (Public) 274  
hex\_chord1() (Public) 728  
hex\_chord2() (Public) 728  
hex\_chord3() (Public) 728  
hex\_coordinates() (Public) 730  
hex\_end() (Public) 714  
hex\_in\_shrink\_set() (Private) 206

hex\_index() (Private) 552  
hex\_knowing\_node() (Public) 211  
hex\_list() (Public) 339, 609, 714, 715  
hex\_point1() (Public) 728  
hex\_point2() (Public) 728  
hex\_point3() (Public) 728  
hex\_smooth\_controller() (Public) 346  
Hexer (Public, constructor) 332  
Hexer (Public, destructor) 332  
hexes() (Public) 133, 141, 164, 503  
HexKnowingNode (Public, constructor) 339  
HexQuality (Public, constructor) 341  
HexQuality (Public, destructor) 341  
HexQualityController (Public, constructor) 343  
HexQualityController (Public, virtual, destructor) 343  
HexTool (Public, constructor) 345  
HexTool (Public, destructor) 345  
HexToVoid (Public, constructor) 349  
HexToVoid (Public, destructor) 349  
highlight() (Public) 224, 241  
highlight\_color() (Public) 224, 241  
highlight\_handler() (Public, static) 321  
histogram() (Public) 453  
hits() (Public) 392  
hl\_copy\_node() (Public) 621  
hl\_edge\_to\_delete() (Public) 621  
hl\_edge\_to\_save() (Public) 622  
hole\_handler() (Public, static) 283  
hyperhelp() (Public, static) 115  
hyperhelp\_handler() (Public, static) 115

## I

---

i\_coord() (Public) 566, 638  
i\_global() (Public) 566, 638  
i\_j\_k() (Public) 566, 638  
id() (Public) 93, 96, 137, 566, 605, 638, 709, 723  
id\_boundary\_faces() (Public) 335  
id\_internal\_faces() (Public) 335  
id\_proto\_face() (Public) 440  
id\_proto\_neighbor() (Public) 335  
id\_proto\_use() (Public) 440  
if() (Public) 431, 621, 627

# Function Index

ijk\_plane() (Public) 490  
ijk\_type() (Public) 566  
import\_file() (Public, static) 285  
import\_solid\_model() (Public) 309  
imprint() (Public) 312  
imprint\_BODYs() (Public) 56  
improve\_boundary\_star() (Private) 109  
improve\_node\_connectivity() (Private) 105  
in\_list() (Public) 640  
in\_shrink\_set() (Public) 211, 431, 627  
in\_star\_list() (Public) 211, 431, 627  
incompatibility() (Private) 101  
incr\_refStructure\_use\_count() (Public) 268, 463, 469, 490, 496, 499, 505, 563  
incr\_refStructure\_use\_count() (Public, virtual) 473  
increment\_offset() (Private) 438  
increment\_use\_count() (Public) 473  
index() (Public) 274  
info\_flag() (Public) 156  
init() (Private) 528  
init() (Public) 216, 235, 643  
init\_brick\_volume() (Public) 552  
init\_gmem() (Private) 231  
initial\_hex\_array() (Private) 551  
initialize() (Private) 181, 465, 484, 551, 693  
initialize() (Public) 565, 638  
initialize\_corners() (Private) 377  
initialize\_counts() (Private) 369  
initialize\_echo\_stream() (Public, static) 115  
initialize\_face\_parameters() (Private) 373  
initialize\_member\_variables() (Private) 161  
initialize\_neighbors() (Public) 526  
initialize\_view() (Public) 220, 238  
inner\_loops() (Private) 570  
inner\_loops\_match() (Private) 585  
insert\_boundary\_layer\_elements() (Private) 101  
insert\_center\_node() (Private) 671  
insert\_edge\_nodes() (Public) 377  
insert\_edge\_type() (Public) 505  
insert\_face\_nodes() (Private) 687  
insert\_gamma\_edge\_nodes() (Private) 687  
insert\_hex() (Private) 407, 433  
insert\_interior\_nodes() (Public) 377  
insert\_item\_sorted() (Protected) 512  
insert\_link() (Protected) 200, 512  
insert\_link\_first() (Protected) 200, 512  
insert\_pillow() (Private) 207  
insert\_proto\_hex() (Public) 87  
insert\_vertex\_type() (Public) 489  
insert\_wedge() (Private) 425  
inside\_chord\_list() (Public) 741  
inside\_loop() (Public) 525  
inside\_out() (Public) 412  
inside\_out\_with\_vecs() (Private) 695  
instance() (Public) 701  
instance() (Public, static) 55, 156, 215, 277, 287, 299, 309, 397, 405, 479, 681, 765  
intelligently\_drive() (Public) 737  
internal\_error() (Public) 156  
internal\_get\_input() (Private) 682  
internal\_mark() (Public) 134  
internal\_zoom() (Private, static) 321  
interpolate\_size() (Public) 119, 607  
intersect() (Protected) 678  
intersect() (Public) 172, 311, 392  
intersect\_2D\_polygon() (Private) 391  
intersect\_3D\_plane() (Private) 391  
intersect\_edge() (Private) 391  
intersect\_edge() (Public) 84  
intersect\_face() (Private) 391  
intersect\_face() (Public) 84  
intersect\_node\_merge() (Public) 84  
intersect\_node\_smooth() (Private) 332  
intersect\_status() (Public) 392  
interval\_bound() (Public) 616  
interval\_changed() (Public) 469  
interval\_changed() (Public, virtual) 473  
interval\_changed\_flag() (Public) 485  
interval\_count() (Public) 472  
interval\_hard() (Public) 466  
interval\_hard() (Public, virtual) 472  
interval\_has\_changed() (Public) 485  
interval\_size() (Public) 472  
interval\_size\_metric() (Protected) 377  
IntervalLinearProgram (Public, constructor)

# Function Index

351  
IntervalLinearProgram (Public, destructor) 351  
intervals\_multiple\_link() (Private) 678  
intervals\_single\_link() (Private) 678  
intervals\_to\_sweep() (Protected, virtual)  
    676  
invalid\_chord() (Public) 716  
invalid\_points() (Public) 747  
invalid\_seam\_neighbors() (Private) 330,  
    348  
invalid\_seam\_neighbors() (Public, virtual)  
    84  
invalid\_sheet() (Public) 746  
inverse() (Public) 153  
inverse\_area() (Public) 141  
invert\_neighbors() (Private) 740  
invert\_neighbors() (Public) 527  
invert\_neighbors() (Public, static) 521  
invert\_sheetschords() (Private) 740  
is\_a\_toggle() (Protected, static) 116  
is\_a\_wedge() (Public) 728  
is\_adjacent() (Public) 710  
is\_at\_beginning() (Public) 200  
is\_at\_end() (Public) 200  
is\_auto\_centering\_on() (Public) 218, 236  
is\_axis\_on() (Public) 218, 236  
is\_blind() (Public) 714  
is\_border\_on() (Public) 218, 236  
is\_cell\_index() (Public) 660  
is\_center() (Public) 163  
is\_center\_node() (Public) 660  
is\_copied() (Public) 660  
is\_cross() (Public) 525  
is\_data\_valid() (Protected) 453  
is\_distance() (Public) 660  
is\_doublet() (Public) 527, 731  
is\_genesis\_id() (Public) 660  
is\_geometry\_posted() (Public) 216, 235  
is\_geometry\_sizing() (Public) 660  
is\_hardpoint\_labeling\_on() (Public) 217,  
    235  
is\_hardpoint\_posted() (Public) 216, 235  
is\_hex\_knowing() (Public) 660  
is\_in\_order() (Public) 467  
is\_inny\_or\_outy() (Private) 694  
is\_labeled() (Public) 297  
is\_labeled() (Public, virtual) 137, 472  
is\_labeling\_on() (Public) 217, 235  
is\_laplace() (Public) 660  
is\_layer() (Public) 660  
is\_leaf() (Public) 667  
is\_loop\_point() (Public) 528  
is\_lp\_edge() (Public) 660  
is\_mapping\_face() (Public) 660  
is\_marked() (Public) 714, 728  
is\_mesh\_posted() (Public) 216, 235  
is\_meshed() (Public) 495  
is\_meshed() (Public, virtual) 472  
is\_meshEdge\_labeling\_on() (Public) 217,  
    218, 235, 236  
is\_meshEdge\_posted() (Public) 216, 217,  
    235  
is\_meshFace\_labeling\_on() (Public) 217,  
    218, 235, 236  
is\_meshFace\_posted() (Public) 216, 217,  
    235  
is\_meshHex\_labeling\_on() (Public) 217,  
    218, 235, 236  
is\_meshHex\_posted() (Public) 216, 217, 235  
is\_meshNode\_labeling\_on() (Public) 217,  
    218, 235, 236  
is\_meshNode\_posted() (Public) 216, 217,  
    235  
is\_morph() (Public) 660  
is\_morphable() (Public) 585  
is\_neighbor() (Public) 527  
is\_node\_hard\_line() (Public) 660  
is\_nodes\_boundary\_mark\_set() (Public)  
    163  
is\_nodeset\_posted() (Public) 216, 217, 235  
is\_normal() (Public) 660  
is\_owned() (Private) 331  
is\_owned() (Public) 274  
is\_parent\_EDGE() (Public) 468  
is\_parent\_VERTEX() (Public) 498  
is\_paver() (Public) 660  
is\_peri\_ints() (Public) 571  
is\_periodic() (Public) 490, 639  
is\_perspective\_on() (Public) 218, 236  
is\_pillow() (Public) 660  
is\_projection() (Public) 660  
is\_refBody\_labeling\_on() (Public) 217,

# Function Index

218, 235  
is\_refEdge\_labeling\_on() (Public) 217, 218, 235, 236  
is\_refFace\_labeling\_on() (Public) 217, 218, 235, 236  
is\_refVertex\_labeling\_on() (Public) 217, 218, 235, 236  
is\_refVertex\_posted() (Public) 216, 235  
is\_refVolume\_labeling\_on() (Public) 217, 218, 235  
is\_set() (Public) 593  
is\_side\_node() (Private) 696  
is\_sideset\_posted() (Public) 216, 217, 235  
is\_size\_set() (Public) 491  
is\_sizing() (Public) 660  
is\_strider() (Public) 660  
is\_sub\_map() (Public) 660  
is\_tipton() (Public) 660  
is\_triangle() (Public) 141  
is\_u\_v\_space() (Public) 660  
is\_vertex\_off() (Public) 489, 593  
is\_vertex\_set() (Public) 489  
is\_visible() (Public) 297, 462, 466, 485, 495, 502  
is\_visible() (Public, virtual) 137, 472, 741  
is\_wedge() (Public) 144, 535, 558, 716  
is\_wedge() (Public, static) 536, 558  
is\_weight() (Public) 660  
is\_whisker\_face() (Public) 660  
is\_whisker\_hex() (Public) 660  
isAutoClearingOn() (Public) 218, 236  
isoparametric\_boundary\_smooth() (Public) 580  
isoparametric\_smooth() (Private) 579  
isoparametric\_smooth() (Public) 345  
isPoly() (Public) 701

## J

---

j\_coord() (Public) 566, 638  
j\_global() (Public) 566, 638  
join() (Public) 371, 474  
join\_blind\_knife() (Public) 715  
join\_chords() (Private) 433  
join\_chords() (Public) 715, 758  
join\_first() (Public, static) 757

join\_hooks() (Private) 371  
join\_neighbors() (Private) 433  
join\_self\_chords() (Public) 521, 715  
join\_sheet() (Public) 743  
join\_to\_form\_wedges() (Private) 754  
join\_to\_form\_wedges() (Public) 757  
journal\_filename() (Public) 682  
journal\_stream() (Private) 682  
just\_mesh() (Public) 577

## K

---

k\_global() (Public) 566, 638  
key() (Public) 481  
keyword\_help() (Public, static) 115  
knife() (Public) 274  
knife\_handler() (Public, static) 389

## L

---

label() (Public) 462, 467, 487, 498, 504  
label\_boundary() (Private) 332  
label\_handler() (Public, static) 321  
laplacian\_smooth() (Private) 579  
laplacian\_smooth() (Public) 345  
laplacian\_smooth\_controller() (Private) 579  
last() (Public) 200  
last\_hex() (Public) 715  
last\_on\_all() (Public) 730  
last\_point() (Public) 520  
last\_transf() (Public) 461  
layer() (Public) 613  
layer\_smooth() (Public) 505  
left\_4\_3043() (Private) 107  
left\_4\_43545() (Private) 107  
left\_5\_34445() (Private) 108  
left\_5\_34454() (Private) 107  
left\_nodes() (Public) 95  
left\_vertex() (Public) 96  
leftBL() (Public) 466  
length() (Public) 134, 184, 468  
length\_squared() (Public) 134  
length\_weighted\_laplacian\_smooth() (Public) 580  
lengthen\_list() (Protected) 64

# Function Index

lengthen\_stack() (Private) 181  
level() (Public) 439, 625  
line\_handler() (Public, static) 283  
line\_width() (Public) 223, 240  
linear\_sizing\_function() (Private) 484  
LinearProgram (Public, constructor) 357  
LinearProgram (Public, virtual, destructor) 357  
lines\_intersect() (Public) 134  
link\_ref\_edges() (Public) 313  
link\_ref\_faces() (Public) 313  
link\_ref\_vertices() (Public) 313  
link\_ref\_volumes() (Public) 313  
list() (Public) 94, 254, 416, 453, 542, 558, 710  
list() (Public, virtual) 297  
list\_2cells() (Public) 747  
list\_body\_entity() (Public) 405  
list\_chord\_entity() (Public) 405  
list\_curve\_entity() (Public) 405  
list\_doublet\_entity() (Public) 405  
list\_doublets() (Public) 747  
list\_entities() (Public, static) 403  
list\_genesis\_entity() (Public) 405  
list\_group\_entity() (Public) 405  
list\_handler() (Private, static) 451  
list\_hex\_entity() (Public) 405  
list\_keywords() (Public, static) 115  
list\_mesh\_edge() (Public) 405  
list\_mesh\_face() (Public) 405  
list\_node\_entity() (Public) 405  
list\_ref\_entity() (Public) 405  
list\_refentity\_names() (Public) 479  
list\_single\_curve\_entity() (Public) 405  
list\_surface\_entity() (Public) 405  
list\_total\_entities() (Public) 405  
list\_vertex\_entity() (Public) 405  
list\_view() (Public) 224, 241  
list\_volume\_entity() (Public) 405  
list\_whisker\_hex\_entity() (Public) 405  
list\_whisker\_sheet\_entity() (Public) 405  
load\_morph\_data() (Public) 484  
load\_sizing\_data() (Public) 484  
local\_id() (Public) 615  
local\_print\_hexes() (Private) 714  
locate\_nodes() (Private) 570  
locate\_on\_sheet() (Public) 522, 717  
look\_at\_entity() (Public) 333  
look\_at\_void\_center() (Public) 333  
Loop (Public, constructor) 363  
Loop (Public, destructor) 363  
loop\_angle() (Public) 724  
loop\_color() (Private) 232  
loop\_edge() (Public) 528  
loop\_face() (Public) 747  
LOOP\_is\_external() (Public) 310  
loop\_me() (Public) 759  
loop\_my\_mesh() (Public) 759  
loop\_node() (Public) 640  
loop\_points() (Public) 522  
loop\_position() (Public) 742  
loop\_self\_intersection() (Private) 433  
loop\_vector() (Public) 745  
loop\_volume() (Public, static) 389  
LoopCollapser (Public, constructor) 365  
LoopInterval (Private, constructor) 369  
LoopJoiner (Public, constructor) 371  
lwl\_boundary\_adjustment() (Public) 580  
lwl\_boundary\_smooth() (Public) 580

---

## M

make() (Public) 148, 149  
make\_acis\_face\_with\_surf() (Public) 58  
make\_acis\_vertex() (Public) 58  
make\_block() (Private) 552  
make\_connect\_to() (Public) 280  
make\_curve() (Public) 58  
make\_elliptical\_curve() (Public) 58  
make\_face() (Public) 57  
make\_face\_with\_surf() (Public) 58  
make\_fastq\_spiral() (Private) 61  
make\_from\_hex() (Public) 148  
make\_node() (Public) 377  
make\_node\_loop() (Public) 571  
make\_parabolic\_curve() (Public) 58  
make\_planar\_quad\_BODY() (Public) 57  
make\_second\_broken\_sheet() (Private) 433, 740  
make\_simple\_hole() (Public) 487  
make\_spline\_curve\_with\_nodes() (Public) 58  
make\_spline\_curve\_with\_surf() (Public)

# Function Index

58  
make\_vertex() (Public) 58  
make\_vertex\_list() (Private) 678  
map\_volume() (Public) 688  
mapping\_requirements() (Public) 571  
mapping\_vertex() (Public) 499  
MappingFace (Public, constructor) 373  
MappingFace (Public, destructor) 374  
MapToolSupport (Public, constructor) 377  
MapToolSupport (Public, virtual, destructor)  
    377  
mark() (Private) 315  
mark\_as\_drawn() (Public) 527  
mark\_boundary\_mesh\_edges() (Protected)  
    580  
mark\_existing\_edges() (Protected) 677  
mark\_faces() (Private) 688  
mark\_hard\_line\_ang() (Public) 581  
mark\_list\_entries() (Private) 678  
mark\_my\_nodes() (Public) 254  
mark\_new\_edges() (Protected) 677  
mark\_node\_vertex() (Private) 688  
mark\_nodes() (Private) 688  
marked() (Public) 134, 141, 162, 468, 469,  
    475, 525, 536  
marked\_neighbors() (Public) 163  
marked\_neighbors\_and\_edges() (Public)  
    163  
master\_ref\_edge() (Public) 195  
master\_ref\_face() (Public) 588  
match\_and\_merge\_lists() (Private) 585  
match\_boundaries() (Private) 585  
match\_boundaries\_cubitedge() (Private)  
    585  
match\_boundaries\_one\_vertex() (Private)  
    585  
match\_entity() (Public) 528  
match\_inner\_loops() (Private) 585  
match\_intervals() (Public) 495  
match\_refEdge\_ref\_vertices() (Public)  
    311  
match\_surface\_intervals() (Private, static)  
    77  
max\_cell\_x() (Public) 191  
max\_cell\_y() (Public) 191  
max\_cell\_z() (Public) 191  
max\_edge\_id() (Public) 399  
max\_element() (Public) 455  
max\_face\_id() (Public) 399  
max\_hex\_id() (Public) 399  
max\_node\_id() (Public) 399  
max\_value() (Public) 455  
max\_variable\_column() (Public) 351  
max\_whisker\_sheet\_id() (Public) 399  
max\_x() (Public) 705  
max\_y() (Public) 706  
max\_z() (Public) 706  
maxDouble() (Public) 378  
maximize\_window() (Public) 221, 238  
maximum() (Public) 125  
maximum\_allocated\_memory() (Public, stat-  
    ic) 63  
maximum\_dimension() (Public) 495  
maximum\_id() (Public) 399  
mean() (Public) 455  
MemoryBlock (Public, constructor) 381  
MemoryBlock (Public, destructor) 381  
MemoryManager (Private, constructor) 383  
MemoryManager (Public, constructor) 384  
MemoryManager (Public, destructor) 384  
merg\_hard\_line\_edges() (Public) 582  
merge() (Private) 735  
merge() (Public) 468, 488, 498  
merge\_all\_handler() (Public, static) 303  
merge\_all\_refbodies() (Public) 313  
merge\_all\_refedges() (Public) 313  
merge\_all\_reffaces() (Public) 313  
merge\_all\_refvertices() (Public) 313  
merge\_body\_handler() (Public, static) 303  
merge\_curve\_handler() (Public, static) 303  
merge\_edge() (Public) 134  
merge\_entity\_names() (Public) 472  
merge\_face() (Public) 141, 724  
merge\_face() (Public, static) 724  
merge\_node() (Public) 164  
merge\_node() (Public, static) 760  
merge\_node\_only() (Public) 164  
merge\_owner() (Public, virtual) 67, 71, 73  
merge\_refbodies() (Public) 313  
merge\_refedge\_use\_counts() (Private) 315  
merge\_refedges() (Public) 313  
merge\_refentity\_names() (Public) 479

# Function Index

merge\_refface\_use\_counts() (Private) 315  
merge\_reffaces() (Public) 313  
merge\_refStructure\_use\_count() (Public) 311  
merge\_refvertex\_use\_counts() (Private) 315  
merge\_refvertices() (Public) 313  
merge\_regions() (Private, static) 283  
merge\_subdomain\_boundaries() (Protected) 419  
merge\_surface\_handler() (Public, static) 303  
merge\_unique() (Public) 200, 511  
merged\_sheet() (Public) 741  
merged\_sheet\_radius() (Public, static) 741  
mesh\_attribute\_handler() (Public, static) 77  
mesh\_color() (Public) 469, 485, 502  
mesh\_color() (Public, virtual) 472  
mesh\_edge\_label\_color() (Public) 222, 223, 234  
mesh\_entity() (Public) 527  
mesh\_entity\_list() (Public) 398  
mesh\_entity\_ptr() (Public) 263  
mesh\_face\_label\_color() (Public) 223, 234  
mesh\_handler() (Public, static) 389  
mesh\_interior\_edges() (Private) 671  
mesh\_is\_visible() (Public) 462, 485, 495, 502  
mesh\_is\_visible() (Public, virtual) 472  
mesh\_left() (Public) 249  
mesh\_left\_and\_right() (Public) 249  
mesh\_linking\_faces() (Protected) 676  
mesh\_me() (Public) 195, 463, 467, 487, 495, 504, 587  
mesh\_me() (Public, virtual) 472  
mesh\_my\_faces() (Public) 504  
mesh\_node\_label\_color() (Public) 222, 223, 234  
mesh\_normally() (Public) 249  
mesh\_right() (Public) 249  
mesh\_scheme() (Public) 495  
mesh\_scheme() (Public, virtual) 473  
mesh\_scheme\_name() (Public) 473, 495  
mesh\_scheme\_name() (Public, virtual) 473  
mesh\_source\_faces() (Protected) 677  
mesh\_sub\_volume() (Private) 695  
mesh\_subregions() (Private) 671  
mesh\_tool() (Public) 504  
mesh\_virtual\_edge() (Public, virtual) 419, 581  
mesh\_wedge() (Private) 101  
MeshEntity (Public, constructor) 387  
MeshEntity (Public, virtual, destructor) 387  
meshing\_algorithm\_name() (Public) 509, 595, 665  
meshing\_algorithm\_name() (Public, virtual) 675  
MeshIntersect (Public, constructor) 392  
MeshIntersect (Public, destructor) 392  
MeshTool (Public, constructor) 395  
MeshTool (Public, destructor) 395  
message() (Public) 215, 234  
MessageFlag (Public, constructor) 155  
MessageFlag (Public, destructor) 155  
metric() (Public) 341, 447  
metric() (Public, virtual) 263  
metric\_name() (Public) 343, 449  
metric\_name() (Public, static) 341, 447  
metric\_name() (Public, virtual) 453  
mid\_point() (Public) 468  
min\_cell\_x() (Public) 191  
min\_cell\_y() (Public) 191  
min\_cell\_z() (Public) 191  
min\_element() (Public) 455  
min\_value() (Public) 455  
min\_x() (Public) 705  
min\_y() (Public) 705  
min\_z() (Public) 705  
minimal\_foreign\_entities() (Private) 307  
minimum() (Public) 125  
mk\_plane\_with\_points() (Public) 171  
Model (Protected, constructor) 397  
Model (Public, destructor) 397  
model\_segment() (Public) 221, 238  
ModelViewer (Protected, constructor) 405  
ModelViewer (Public, destructor) 405  
morph\_attribute\_handler() (Private, static) 77  
Mouse (Public, constructor) 408  
Mouse (Public, destructor) 408  
mouse\_2D() (Private) 407

# Function Index

mouse\_2D\_setup() (Private) 407  
mouse\_3D() (Private) 407  
mouse\_3D\_setup() (Private) 407  
mouse\_xforms() (Public) 215, 234, 408  
mouse\_xforms\_3dSegment() (Public, static)  
    321  
mouse\_xforms\_screenSegment() (Public,  
    static) 321  
move() (friend) 465  
move() (Public) 416, 463, 498  
move() (Public, virtual) 297, 473  
move\_between\_items() (Protected) 201  
move\_body() (Public) 312  
move\_bounding\_nodes\_to\_surface() (Pub-  
    lic) 487  
move\_sheetschordpoints() (Public) 747  
move\_start\_to\_twin() (Public) 716  
move\_to() (Public) 416  
move\_to() (Public, virtual) 297  
move\_to\_and\_remove\_item() (Protected)  
    201  
move\_to\_and\_remove\_item\_sorted() (Pro-  
    tected) 512  
move\_to\_item() (Protected) 201  
move\_to\_item\_sorted() (Protected) 512  
move\_to\_owner() (Public) 162  
move\_to\_surface() (Public) 487  
moveRefVertex() (Private) 465  
moveToCurve() (Public) 468  
MuseOutput (Public, constructor) 409  
my\_orphan() (Public) 522

## N

---

name() (Public) 71  
name() (Public, virtual) 277, 278, 279, 280  
name\_to\_index() (Protected) 453  
narrow\_transition() (Private) 110  
nbFlag() (Public) 498  
nearest\_node() (Private) 266  
need\_hex\_element\_center\_nodes() (Pub-  
    lic) 505  
need\_mesh\_edge\_center\_nodes() (Private)  
    301  
need\_mesh\_edge\_center\_nodes() (Public)  
    487, 505

need\_mesh\_face\_center\_nodes() (Private)  
    301  
need\_mesh\_face\_center\_nodes() (Public)  
    488, 505  
need\_to\_update() (Private) 233  
need\_to\_update() (Public) 221, 239, 453  
neg\_i\_ptr() (Public) 640  
neg\_j\_ptr() (Public) 640  
neg\_k\_ptr() (Public) 640  
neighbor() (Public) 290, 439  
neighbor\_case() (Private) 438  
neighbor\_hex() (Public) 290  
neighbor\_marked() (Public) 525  
neighboring\_hex() (Public) 717  
neighboring\_point() (Public) 526  
neighboring\_point\_index() (Public) 526  
neighbors\_connected() (Public) 438  
neighbors\_drawn() (Private) 529  
new\_blind\_chord\_pair() (Public) 744  
new\_blind\_points() (Public) 729  
new\_chord\_segment() (Private) 232  
new\_current\_blades() (Private) 735  
new\_curve\_segment() (Private) 232  
new\_edge\_segment() (Private) 232  
new\_end\_ref\_vertex() (Public) 468  
new\_extended\_point() (Public) 520  
new\_face\_segment() (Private) 232  
new\_inside\_list() (Public) 746  
new\_level() (Private) 331, 348  
new\_node\_segment() (Private) 232  
new\_nodeset\_segment() (Private) 232  
new\_overlay\_segment() (Private) 232  
new\_paver\_node() (Private) 427  
new\_screen\_segment() (Private) 232  
new\_segment() (Private) 232  
new\_sheet\_loop\_segment() (Private) 232  
new\_sheet\_segment() (Private) 232  
new\_sideset\_segment() (Private) 232  
new\_start\_ref\_vertex() (Public) 468  
new\_surface\_segment() (Private) 232  
new\_temp\_segment() (Private) 232  
new\_text\_segment() (Private) 232  
new\_vertex\_segment() (Private) 232  
next() (Public) 381  
next\_block() (Public) 381  
next\_child\_item() (Public) 668

# Function Index

next\_constraint() (Public) 358  
next\_curve\_sub\_domain\_id() (Public) 398  
next\_dummy() (Public) 351  
next\_edge() (Public, static) 134  
next\_face() (Public, static) 141  
next\_free\_instance() (Public) 224  
next\_hex() (Public, static) 143  
next\_hex\_point() (Public) 709  
next\_item() (Protected) 201  
next\_loop() (Public) 486  
next\_loop\_sheet\_chord() (Public) 744  
next\_mapping\_face() (Public, static) 577  
next\_node() (Private) 266  
next\_node() (Public) 566  
next\_node() (Public, static) 164  
next\_proto\_hex() (Private) 331, 348  
next\_proto\_hex\_from\_face() (Private) 348  
next\_ref\_body\_id() (Public) 398  
next\_ref\_edge\_id() (Public) 398  
next\_ref\_face\_id() (Public) 398  
next\_ref\_group\_id() (Public) 398  
next\_ref\_vertex\_id() (Public) 398  
next\_ref\_volume\_id() (Public) 398  
next\_sheet\_chord() (Public) 743  
next\_surf\_sub\_domain\_id() (Public) 398  
next\_tool\_data() (Public) 659  
no\_more\_tokens() (Public) 681  
no\_test\_edges() (Public) 625  
nodal\_neighbor() (Public) 439  
nodal\_smooth\_controller() (Public) 346  
node() (Public) 192, 411, 498  
node\_cleaned() (Public) 165  
node\_color() (Private) 232  
node\_cross\_another() (Private) 427  
node\_cross\_self() (Private) 427  
node\_distance() (Public) 162, 603  
node\_hex() (Public) 728  
node\_id() (Public) 335  
node\_index() (Private) 411, 552  
node\_is\_ON() (Public) 314  
node\_length\_weighted\_laplacian\_smooth()  
    () (Private) 579  
node\_loop() (Public) 487  
node\_loops() (Public) 486  
node\_marked() (Private) 688  
node\_needed() (Public) 162  
node\_num() (Private) 552  
node\_perm() (Public) 162  
node\_smooth\_last() (Public) 162  
node\_smoothed() (Public) 162  
node\_throw() (Public) 439  
node\_u() (Public) 647  
node\_v() (Public) 647  
node\_x() (Public) 162  
node\_y() (Public) 162  
node\_z() (Public) 162  
NodeHex (Public, constructor) 411  
NodeHex (Public, destructor) 411  
nodes() (Public) 140, 148, 268, 274, 289, 411,  
        415, 466, 486, 503  
nodes() (Public, virtual) 143  
nodes\_boundary() (Public) 503  
nodes\_boundary\_mark() (Public) 163  
nodes\_boundary\_unmark() (Public) 163  
nodes\_inclusive() (Public) 268, 486, 503  
nodes\_splitting\_loop() (Public) 743  
NodeSet (Private, constructor) 416  
NodeSet (Public, constructor) 415  
NodeSet (Public, destructor) 415  
nodeset\_assoc() (Private) 301  
nodeset\_color() (Private) 232  
nodeset\_list() (Public) 299  
normal() (Public) 140, 171, 274  
normal\_at() (Public) 487  
normal\_face() (Public) 623  
normal\_set() (Public) 623  
normal\_valid() (Public) 162  
normal\_vector() (Public) 623, 745  
nullify\_link() (Protected) 200  
num\_attached\_nodes() (Public) 639  
num\_attributes() (Public) 254  
num\_cells() (Public) 705  
num\_cells\_x() (Public) 705  
num\_cells\_y() (Public) 705  
num\_cells\_z() (Public) 705  
num\_children() (Public) 667  
num\_cols() (Public) 153  
num\_coordinates() (Public) 254  
num\_coordinates() (Public, virtual) 297  
num\_doublets() (Public) 755  
num\_dummies() (Public) 351  
num\_edges() (Public) 192, 351

# Function Index

num\_element\_blocks() (Public) 287, 299, 765  
num\_elements() (Public) 287, 299, 766  
num\_geometry\_entities() (Public) 253, 415, 541  
num\_geometry\_entities() (Public, virtual) 298  
num\_inside\_chords() (Public) 742  
num\_inside\_lists() (Public) 746  
num\_mesh\_entities() (Public) 253, 415, 541  
num\_mesh\_entities() (Public, virtual, virtual) 297  
num\_neighbors() (Public) 526  
num\_nodes() (Public) 192, 288, 299, 766  
num\_nodes\_per\_element() (Public) 254  
num\_rows() (Public) 153  
num\_sheet\_chords() (Public) 742  
num\_sides() (Public) 374  
num\_to\_pos() (Private) 552  
num\_wedges() (Public) 742  
num\_whisker\_faces() (Public) 742  
number\_columns() (Public) 176, 177  
number\_deleted() (Public) 755  
number\_edge\_uses() (Public, static) 252  
number\_edges() (Public) 163  
number\_edges() (Public, static) 134  
number\_edges\_on\_surface() (Private) 109  
number\_elements() (Public) 94  
number\_external\_proto\_hexes() (Public) 84  
number\_face\_uses() (Public) 141  
number\_face\_uses() (Public, static) 274  
number\_faces() (Public) 133  
number\_faces() (Public, static) 141  
number\_grid\_cells() (Public) 324  
number\_grid\_cells\_x() (Public) 324  
number\_grid\_cells\_y() (Public) 324  
number\_grid\_cells\_z() (Public) 324  
number\_hexes() (Public, static) 143  
number\_items() (Public) 129  
number\_knife\_uses() (Public, static) 148  
number\_layers() (Public) 177  
number\_marked\_edges() (Public) 163  
number\_nodes() (Public, static) 164  
number\_of\_debug\_flags() (Public) 156  
number\_of\_edge\_loops() (Public) 486  
number\_of\_metrics() (Public) 343, 449  
number\_of\_metrics() (Public, static) 341, 447  
number\_of\_metrics() (Public, virtual) 453  
number\_of\_processors() (Public) 420  
number\_of\_samples() (Public) 455  
number\_processable\_proto\_hexes() (Public) 84  
number\_proto\_hexes() (Public) 84  
number\_proto\_hexes\_at\_level() (Public) 84  
number\_rows() (Public) 176, 177  
number\_whisker\_hexes() (Public) 715  
number\_whisker\_hexes() (Public, static) 730  
number\_whisker\_sheets() (Public, static) 746  
numbered\_hex\_vector() (Public) 716

---

## O

objective\_set() (Public) 352, 616  
offset() (Public) 274  
ok\_to\_collapse() (Public) 165  
on\_boundary() (Public) 141, 162, 439  
on\_boundary() (Public, virtual) 274, 439  
on\_loop() (Public) 525, 535, 558  
on\_loop() (Public, static) 535, 558  
on\_meshed\_boundary() (Private) 207  
on\_new\_blade\_side() (Private, static) 147  
one\_edge\_boundary\_layer() (Private) 102  
one\_middle\_node() (Private) 108  
one\_row\_transition() (Private) 109  
one\_to\_three() (Private) 107  
open\_crack() (Private, static) 147  
open\_crack\_break\_loop\_weaving() (Private, static) 148  
open\_crack\_edge() (Private, static) 147  
open\_crack\_fix\_chord\_weaving() (Private, static) 148  
open\_face() (Private) 107  
open\_file() (Public) 267  
open\_journal\_stream() (Public) 681  
open\_recording\_stream() (Public) 681  
open\_surface\_crack() (Public, static) 149  
operator\_delete() (Public) 384  
operator\_new() (Public) 384

# Function Index

opposite\_edge() (Public) 140  
opposite\_edge\_use() (Public) 274  
opposite\_face() (Public) 290, 412  
opposite\_face() (Public, virtual) 144  
opposite\_face\_use() (Public) 290, 412  
opposite\_face\_use() (Public, virtual) 144  
opposite\_node() (Public) 140  
opposite\_nodes() (Public) 431, 555, 627  
opposite\_point() (Public) 526, 557, 710  
option() (Public) 71  
order\_corners() (Private) 577  
order\_side\_chords() (Private) 735  
ordered\_edges() (Public) 163  
ordered\_faces() (Public) 163  
ordered\_neighbors() (Private) 108  
ordered\_node\_array() (Public) 133, 134, 140, 290, 411  
ordered\_node\_array() (Public, virtual) 143, 387  
ordered\_nodes() (Public) 566, 638  
ordered\_on\_boundary() (Private) 110  
orderedNodes() (Private) 300  
orientation\_known() (Public) 525  
orphan\_sheet\_chord\_1() (Public) 714  
orphan\_sheet\_chord\_2() (Public) 714  
other\_body() (Private) 551  
other\_body() (Public) 552  
other\_cell() (Public) 710  
other\_chord() (Public) 731  
other\_doublet\_point() (Public) 527  
other\_face() (Public) 133  
other\_face\_node() (Public) 413  
other\_faces() (Public) 133  
other\_node() (Public) 134, 164  
other\_nodes() (Public) 611  
other\_points() (Public) 528  
other\_sheet() (Public) 715  
other\_sheet\_chord() (Public) 522, 715, 729  
other\_two\_chords() (Public) 730  
other\_vertex() (Public) 467  
out\_face\_use\_id() (Public) 335  
output() (Public) 155  
output\_debug\_information() (Public) 157  
output\_logging\_information() (Public) 157  
output\_summary() (Private) 287, 300, 765  
outside\_loop() (Public) 525  
outside\_source\_edges() (Protected) 678  
overlay() (Public) 222, 240  
overlay\_color() (Private) 232  
overlay\_edges() (Private) 231  
overlay\_edges() (Public) 149  
overlay\_face() (Public) 239  
overlay\_label() (Public) 134, 140, 143, 163, 223, 240  
overlay\_line() (Public) 239  
overlay\_mode() (Public) 222, 239  
overlay\_point() (Public) 239  
overlay\_text() (Public) 239  
overlay\_vec() (Public) 222, 240  
owner() (Public) 134, 141, 144, 162, 565, 638  
owner() (Public, virtual) 387  
owns\_this\_loop() (Public) 504

---

**P**

pack\_nodes() (Public, static) 164  
pan\_cursor() (Public) 220, 238  
pan\_handler() (Public, static) 321  
pan\_view() (Public) 220, 238  
parallel\_pave() (Public) 419  
ParallelMeshTool (Public, constructor) 419  
ParallelMeshTool (Public, destructor) 419  
parameter\_at() (Private) 465  
parent() (Public) 461, 466, 485, 497, 502, 565, 593, 638, 699  
parent\_acis\_faces() (Public) 488  
parent\_item() (Public) 668  
parent\_on\_top() (Public) 466  
parentEDGES() (Public) 468  
parents() (Public) 73  
parentVERTEXs() (Public) 498  
parse() (Public) 319  
parse\_command\_options() (Private) 682  
parse\_input\_line() (Public) 681  
part\_number() (Public) 254  
partner() (Public) 275  
past\_end() (Private) 561  
pause() (Public, static) 403  
Paver (Public, constructor) 428  
Paver (Public, destructor) 428  
perform\_interface() (Public) 681

# Function Index

period\_u() (Public) 490  
period\_v() (Public) 490  
periodic\_interval() (Public) 491  
periodic\_u() (Public) 489  
periodic\_v() (Public) 489  
permute\_handler() (Public, static) 389  
permute\_state\_list() (Public) 756  
perspective\_angle() (Public) 219, 237  
pick() (Public) 224, 241  
pick\_3\_corners() (Private) 575  
pick\_4\_corners() (Private) 576  
pick\_corners() (Private) 575  
pick\_handler() (Public, static) 321  
pick\_meshed\_corners() (Private) 575, 577  
pick\_ref\_body() (Public) 224, 241  
pick\_ref\_volume() (Public) 224, 241  
pict\_animate\_init() (Public) 238  
pict\_animate\_snap() (Public) 238  
pillow() (Public) 434  
pillow\_doublets() (Public) 208  
pillow\_handler() (Public, static) 389  
pillow\_node() (Public) 211  
pillow\_through\_cells() (Private) 754  
PillowNode (Public, constructor) 431  
PillowNode (Public, destructor) 431  
PillowSheet (Public, constructor) 434  
place\_initial\_nodes() (Private) 573  
place\_node\_at\_centroid() (Public) 192, 706  
plane\_judgement() (Protected) 678  
playback() (Public, static) 285  
point\_A() (Public) 89, 536  
point\_a() (Public) 557  
point\_along\_edge\_from() (Public) 96  
point\_at\_arc\_length() (Public) 468  
point\_B() (Public) 89, 536  
point\_b() (Public) 557  
point\_C() (Public) 557  
point\_c() (Public) 557  
point\_coord() (Public) 525  
point\_d() (Public) 557  
point\_handler() (Public, static) 283  
point\_in\_face() (Private) 266  
point\_in\_polygon() (Public) 192  
point\_in\_polygons() (Public) 192  
point\_index() (Private) 409  
points\_sheet\_chord() (Public) 527  
pop\_queue() (Public) 458  
populate\_line\_list() (Private, static) 284  
populate\_nodal\_sizing\_data() (Public) 268  
populate\_point\_list() (Private, static) 284  
pos\_i\_ptr() (Public) 640  
pos\_j\_ptr() (Public) 640  
pos\_k\_ptr() (Public) 640  
position\_flag() (Public, static) 755  
position\_from\_u\_v() (Public) 488  
position\_physical\_weave() (Public) 758  
position\_point\_list() (Public) 522, 710  
post\_create\_hex\_boundary\_intersects() (Private) 330, 348  
post\_create\_hex\_boundary\_intersects() (Public, virtual) 84  
post\_create\_hex\_graphics() (Private) 330, 348  
post\_create\_hex\_graphics() (Public, virtual) 84  
post\_create\_hex\_smooth\_boundary() (Private) 330, 348  
post\_create\_hex\_smooth\_boundary() (Public, virtual) 84  
post\_create\_hex\_whiskerweave() (Private) 330, 348  
post\_create\_hex\_whiskerweave() (Public, virtual) 84  
post\_seam\_graphics() (Private) 348  
post\_seam\_graphics() (Public) 333  
post\_seam\_graphics() (Public, virtual) 84  
pre\_mesh() (Protected) 676  
pre\_seam\_graphics() (Private) 348  
pre\_seam\_graphics() (Public) 333  
pre\_seam\_graphics() (Public, virtual) 84  
precompute\_feature\_size() (Public) 307  
prepare\_bounding\_curves() (Private) 266  
prev\_item() (Protected) 201  
previous\_interval() (Public) 469  
previous\_state() (Public) 438  
primal\_flag() (Public, static) 755  
primal\_handler() (Public, static) 389  
primitive\_handler() (Public, static) 303  
print() (Public) 134, 171, 455, 535  
print\_cross() (Public) 517

# Function Index

print\_debug() (Public) 156  
print\_diagnostic() (Public) 156  
print\_edges() (Public) 141  
print\_entity\_description() (Public) 526  
print\_error() (Public) 156  
print\_hexes() (Public) 715  
print\_info() (Public) 156  
print\_loops() (Private) 428  
print\_matrix() (Public) 153  
print\_neighbors() (Public) 526  
print\_points() (Public) 520, 558, 715  
print\_solution() (Public, static) 518  
print\_this() (Public) 520  
print\_valid\_names() (Protected) 453  
print\_voxel\_contents() (Public) 192  
print\_warning() (Public) 156  
prioritize\_active\_faces() (Private) 595  
prioritize\_active\_faces() (Protected) 676  
prism() (Public) 310  
problem\_report\_handler() (Public, static) 403  
process() (Private, static) 283  
process\_barset() (Private, static) 283  
process\_hardpoint() (Private) 424  
process\_hole() (Private, static) 283  
process\_loop() (Private) 424  
process\_mesh() (Private) 423  
process\_quality\_scheme() (Private, static) 451  
process\_state\_intersect() (Private) 331  
process\_state\_no\_sides() (Public) 85  
process\_state\_open\_corner() (Private) 331, 349  
process\_state\_open\_corner() (Public, virtual) 85  
process\_state\_perturb\_boundary() (Private) 331  
process\_state\_seam\_case\_4() (Public) 85  
process\_state\_seam\_case\_5() (Public) 85  
process\_state\_seam\_case\_6() (Private) 331, 349  
process\_state\_seam\_case\_6() (Public, virtual) 85  
process\_state\_sides\_0() (Public) 85  
process\_state\_sides\_01() (Public) 85  
process\_state\_sides\_012() (Public) 85  
process\_state\_sides\_0123() (Public) 85  
process\_state\_sides\_01234() (Public) 85  
process\_state\_sides\_02() (Public) 85  
process\_state\_success() (Public) 439  
process\_state\_unknown() (Private) 331, 349  
process\_state\_unknown() (Public, virtual) 85  
process\_state\_unusable() (Private) 331  
process\_state\_voidface() (Private) 349  
process\_toggle() (Protected, static) 116  
processors() (Public) 490  
project\_corner() (Private) 425  
project\_node() (Private) 425  
project\_reversal() (Private) 425  
project\_side() (Private) 425  
project\_to\_axis() (Private) 510  
projection\_node() (Public) 631  
projection\_node\_research() (Protected) 678  
projections\_needed() (Private) 679  
prompt\_to\_continue() (Public) 420, 563, 581  
protected\_face() (Private) 109  
proto\_hex\_validity\_check() (Public) 85  
proto\_hexes() (Public) 85  
proto\_id() (Public) 334  
ProtoHex (Public, constructor) 438  
ProtoHex (Public, destructor) 438  
proximity() (Private) 427  
prune\_points() (Public) 607  
ptr\_to\_key() (Public) 481  
pull\_apart\_five() (Private) 108  
put\_at() (Public) 183  
put\_attrib\_gtc\_name() (Public) 59, 60  
put\_into\_cleanup\_list() (Private) 109  
Pyramid (Public, constructor) 443  
Pyramid (Public, destructor) 443  
pyramid() (Public) 310  
PyramidTool (Public, constructor) 445  
PyramidTool (Public, destructor) 445

## Q

---

quad\_points() (Public) 709

# Function Index

quad\_smooth\_controller() (Public) 581  
QuadQuality (Public, constructor) 447  
QuadQuality (Public, destructor) 447  
QuadQualityController (Public, constructor) 449  
QuadQualityController (Public, virtual, destructor) 449  
quads\_per\_processor() (Public) 419, 490  
quality\_controller() (Public) 475  
quality\_handler() (Public, static) 451  
QualityCommands (Private, constructor) 451  
QualityController (Public, constructor) 453  
QualityController (Public, virtual, destructor) 453  
QualitySummary (Public, constructor) 455  
query\_display\_sheets() (Public) 756  
query\_flag() (Public, static) 755  
query\_resolve\_wedges() (Public) 756  
Queue (Public, constructor) 458  
Queue (Public, virtual, destructor) 458  
queue\_get() (Public) 458  
queue\_get\_and\_step() (Public) 458  
QueueNode (Protected, constructor) 457  
QueueNode (Public, destructor) 457  
quit() (Public, static) 283

**R**

rangeSpecifier() (Protected, static) 115  
read\_cross() (Public) 517  
read\_displacements() (Public) 267  
read\_elements() (Private) 266  
read\_free\_mesh() (Public) 267  
read\_geom\_mesh() (Public) 267  
read\_geometry\_file() (Public) 309  
read\_mesh() (Public) 267  
read\_nodes() (Private) 265  
read\_sizing\_function() (Public) 267  
reading\_fastq() (Public) 682  
recalculate\_position() (Public) 521  
recording\_filename() (Public) 682  
recording\_stream() (Private) 682  
rectangle\_constraints() (Public) 374  
rectangle\_constraints\_possible() (Public) 375  
redistribute() (Public) 573

redraw\_physical\_sheet() (Public) 746  
reduce() (Public) 365  
ref\_bodies() (Public) 397  
ref\_body() (Public) 505  
ref\_edge() (Public) 95, 195  
ref\_edges() (Public) 268, 278, 397, 415, 462, 486, 499, 503, 542  
ref\_entities() (Public) 415, 542  
ref\_face() (Public) 374, 420, 587  
ref\_faces() (Public) 397, 415, 462, 468, 503, 542  
ref\_faces() (Public, virtual) 472  
ref\_groups() (Public) 397  
ref\_vertex\_loops() (Public) 486  
ref\_vertices() (Public) 278, 397, 415, 462, 486, 504  
ref\_volume() (Public) 488, 755  
ref\_volume\_list() (Public) 488  
ref\_volumes() (Public) 397, 415, 462  
RefBody (Public, constructor) 461  
RefBody (Public, destructor) 461  
RefEdge (Public, constructor) 465  
RefEdge (Public, destructor) 465  
RefEntity (Public, constructor) 472  
RefEntity (Public, virtual, destructor) 472  
RefEntityName (Protected, constructor) 480  
RefEntityName (Public, destructor) 479  
RefEntityNameMap (Public, constructor) 481  
RefFace (Public, constructor) 484  
RefFace (Public, destructor) 484  
RefGroup (Public, constructor) 495  
RefGroup (Public, destructor) 495  
reflect() (Public) 463  
reflect() (Public, virtual) 473  
reflect\_about\_xaxis() (Public) 742  
reflect\_body() (Public) 312  
reflect\_sheetschord() (Public) 520  
RefVertex (Public, constructor) 497  
RefVertex (Public, destructor) 497  
RefVolume (Public, constructor) 502  
RefVolume (Public, destructor) 502  
region() (Public) 363  
region\_handler() (Public, static) 283  
reinitialize() (Public) 517  
remove() (Public) 253, 398, 415, 541  
remove() (Public, virtual) 297

# Function Index

remove\_blind\_hexes() (Public) 717  
remove\_blind\_sheet\_chord() (Public) 742  
remove\_boundary\_edge() (Public) 334  
remove\_boundary\_face() (Public) 334  
remove\_boundary\_node() (Public) 87  
remove\_center\_node() (Public) 133, 141, 290  
remove\_cubit\_owner() (Public) 60  
remove\_cubit\_owner\_ATTRIB() (Public) 60, 472  
remove\_cubit\_owner\_ATTRIB\_in\_BODY() (Public) 60  
remove\_current() (Public) 549  
remove\_debug\_stream() (Private) 156  
remove\_deleted\_faces() (Public) 86  
remove\_dir\_node() (Public) 639  
remove\_doubles() (Public) 571  
remove\_edge() (Public) 164, 467, 487, 504  
remove\_edge() (Public, virtual) 473  
remove\_edge\_use() (Public) 133  
remove\_edges() (Public) 99  
remove\_ENTITY\_from\_refentities() (Public) 56  
remove\_entity\_name() (Public) 472  
remove\_exterior\_loops() (Private) 585  
remove\_face() (Public) 251, 487, 504  
remove\_face() (Public, virtual) 473  
remove\_face\_use() (Public) 139  
remove\_from\_group\_list() (Public) 475  
remove\_geometry() (Public) 94  
remove\_gridsearch\_nodes() (Private) 570  
remove\_hex() (Public) 274, 504, 729  
remove\_hex() (Public, virtual) 473  
remove\_hex\_seam() (Public) 757  
remove\_internal\_face() (Public) 334  
remove\_knife() (Public) 274  
remove\_list() (Public) 378  
remove\_loop\_face() (Public) 716  
remove\_loop\_nodes() (Public) 639  
remove\_merged\_face\_uses() (Public) 86  
remove\_mesh() (Public) 195, 267, 463, 468, 488, 495, 498, 505, 587  
remove\_mesh() (Public, virtual) 473  
remove\_mesh\_from\_geometry() (Public) 398  
remove\_node() (Public) 192, 325, 467, 487, 499, 504  
remove\_node() (Public, virtual) 473  
remove\_node\_from\_cell() (Private) 324  
remove\_node\_from\_cleanup() (Private) 109  
remove\_node\_from\_smoothing() (Private) 427  
remove\_outer\_nodes() (Public) 570  
remove\_parent() (Public) 461, 466, 485, 497, 502  
remove\_proto\_hex() (Public) 87  
remove\_ref\_entity() (Public) 496  
remove\_refentity\_name() (Public) 479  
remove\_repeated\_entity() (Public) 758  
remove\_sheet() (Public) 398  
remove\_sheet\_chord() (Public) 756  
remove\_sheet\_faces() (Public) 179  
remove\_small\_angles() (Public) 673  
remove\_submap\_temp\_mesh() (Public) 488  
remove\_TD() (Public) 663  
remove\_this\_face() (Private) 107  
removeGeometry() (Public) 416, 542  
removeGeometryFromBoundaryLayers() (Private) 399  
rep\_hex\_flag() (Public, static) 755  
repeated\_entity() (Public) 521  
repeated\_repeated\_hex() (Public) 758  
replace\_chord() (Public) 746  
replace\_edge() (Public) 140  
replace\_face() (Public, virtual) 144  
replace\_node() (Public) 140, 290, 412  
replace\_node() (Public, virtual) 144  
report\_drives() (Private) 736  
report\_plane\_error() (Public) 172  
report\_superdrive() (Private) 736  
reposition\_boundary\_nodes() (Private) 510, 665  
reposition\_node() (Public) 706  
rescale\_angle() (Public) 742  
reset() (Private) 392  
reset() (Public) 125, 181, 199, 398, 455  
reset() (Public, static) 403  
reset\_all() (Public) 300  
reset\_blocks() (Public) 300  
reset\_counter() (Public) 179  
reset\_counters() (Public) 300  
reset\_doublet\_points() (Public) 755  
reset\_drawn\_flags() (Public) 527

# Function Index

reset\_error\_count() (Public) 157  
reset\_global\_list() (Public, static) 164  
reset\_node\_ownership() (Public) 268  
reset\_nodesets() (Public) 299  
reset\_sheetschord\_smooths() (Public) 520  
reset\_sidesets() (Public) 299  
reset\_state() (Public) 86  
reset\_state\_intersect() (Private) 330, 349  
reset\_state\_intersect() (Public, virtual) 86  
reset\_state\_unusable() (Private) 330, 349  
reset\_state\_unusable() (Public, virtual) 86  
reset\_token() (Public) 681  
reset\_window() (Public, static) 757  
resize\_hit\_point\_array() (Private) 391  
resolve\_doublets() (Public) 758  
resolve\_handler() (Public, static) 389  
resolve\_intersections() (Private) 332, 349  
resolve\_intersections() (Public, virtual) 86  
resolve\_knife\_doublet() (Public) 758  
resolve\_one\_wedge() (Public) 756  
resolve\_wedges() (Public) 756  
restore() (Public) 463  
restore() (Public, virtual) 473  
restore\_body() (Public) 312  
retrieve() (Public) 319  
return() (Public) 705, 706  
returnOrderedSideSetNodes() (Private) 300  
reversal() (Private) 569  
reverse() (Private) 693  
reverse() (Public) 200, 511  
reverse\_alignment() (Public) 587  
reverse\_bias() (Public) 249, 467  
reverse\_bias() (Public, virtual) 472  
reverse\_BODY() (Public) 59  
reverse\_handler() (Public, static) 303  
reverse\_inner\_vertices() (Private) 570  
reverse\_normal() (Public) 140  
reverse\_refbodies() (Public) 309  
reversed() (Public) 96  
reversed\_narrow\_surf() (Private) 424  
reversedNodes() (Public) 466  
right\_4\_3043() (Private) 107  
right\_4\_43545() (Private) 107  
right\_5\_34445() (Private) 108  
right\_5\_34454() (Private) 108  
right\_edge\_angle() (Public) 96  
right\_interior\_nodes() (Public) 96  
right\_nodes() (Public) 95  
right\_vertex() (Public) 96  
rightBL() (Public) 466  
root\_item() (Public) 668  
root\_segment() (Public) 221, 239  
rotate() (Public) 219, 237, 463, 742  
rotate() (Public, static) 321  
rotate() (Public, virtual) 473  
rotate\_best() (Private) 332  
rotate\_body() (Public) 312  
rotate\_camera() (Private) 233  
rotate\_general() (Private) 233  
rotate\_general() (Public) 219, 237  
rotate\_lists() (Private) 109  
rotate\_screen() (Private) 233  
rotate\_to\_camera\_vector() (Public) 334  
rotate\_world() (Private) 233  
RotateTool (Public, constructor) 509  
RotateTool (Public, destructor) 509

---

## S

same\_edge() (Public) 558  
same\_face() (Public) 536, 558  
same\_sheets() (Public) 715  
same\_STC\_edge() (Public) 710, 759  
sample\_hex() (Public) 503  
save\_ENTITY\_as\_sat\_file() (Public) 56  
scale() (Public) 463  
scale() (Public, virtual) 473  
scale\_angle() (Public) 742  
scale\_body() (Public) 312  
scheme\_handler() (Public, static) 283  
screen\_color() (Private) 232  
screen\_coord\_x() (Public) 744  
screen\_coord\_y() (Public) 744  
screen\_draw\_to() (Public) 223, 240  
screen\_label() (Public) 223, 241  
screen\_min\_max() (Public) 223, 240  
screen\_move\_to() (Public) 223, 240

# Function Index

screen\_segment() (Public) 221, 238  
SDLList (Public, constructor) 511  
SDLList (Public, destructor) 511  
seam() (Private) 425  
seam\_and\_wedge() (Private) 425  
seam\_loop() (Private) 425  
seam\_near\_nodes() (Private) 425  
second\_chord() (Public) 731  
second\_last\_hex() (Public) 715  
second\_last\_point() (Public) 520  
second\_layer\_depth() (Public) 94  
second\_point() (Public) 520  
seed\_A() (Public) 89  
seed\_B() (Public) 89  
select\_corners() (Private) 671  
select\_list() (Private, static) 295  
self\_destruct() (Private) 695  
self\_destruct() (Public) 571  
self\_intersecting\_chords() (Private) 740  
self\_intersecting\_chords() (Public) 747  
SelfCrossing (Private, constructor) 515  
SelfCrossingLoop (Public, constructor) 517  
sense() (Public) 252, 274  
separate\_voids() (Public) 704  
seperate\_volumes() (Private) 696  
set() (Private) 693  
set() (Public) 93, 153, 163, 613  
set\_3d\_camera\_field() (Private) 233  
set\_active\_instance() (Public) 224  
set\_assembly\_number() (Public) 254  
set\_cleanup\_type() (Private) 111  
set\_constraint\_name() (Public) 358  
set\_copy\_node() (Public) 640  
set\_corners() (Public) 374  
set\_counts() (Private) 576  
set\_cubit\_owner() (Public) 67  
set\_cubit\_owner() (Public, static) 67  
set\_cubit\_owner\_attrib() (Public) 60  
set\_current\_graphics\_status() (Public) 215  
set\_currentEdgeSegment() (Public) 221, 239  
set\_debug\_file() (Public) 156  
set\_debug\_stream() (Private) 155  
set\_density() (Public) 561  
set\_distance() (Protected, static) 580  
set\_edge() (Private) 694  
set\_edge\_bound() (Public) 352  
set\_edge\_type() (Public) 699  
set\_entry() (Public) 358  
set\_even() (Public) 466  
set\_even\_intervals() (Public) 489  
set\_face() (Private) 694  
set\_face\_orientation() (Protected) 676  
set\_first\_index() (Private) 434, 740  
set\_gain() (Public, static) 408  
set\_genesis\_id() (Public) 254  
set\_guiding\_ref\_faces() (Public) 502  
set\_hsr\_algorithm() (Private) 232  
set\_in\_out\_index() (Private) 365  
set\_in\_out\_seed() (Private) 365  
set\_increment() (Public) 63, 245  
set\_index() (Public) 191  
set\_inside\_recurse() (Private, static) 433  
set\_interval\_count() (Public, virtual) 472  
set\_interval\_count\_user() (Public) 469  
set\_interval\_count\_user() (Public, virtual) 472  
set\_interval\_even() (Public) 545  
set\_interval\_size() (Public, virtual) 472  
set\_interval\_size\_user() (Public) 469, 485  
set\_interval\_size\_user() (Public, virtual) 472  
set\_layer\_vector() (Private) 665  
set\_logging\_file() (Public) 156  
set\_lower\_bound() (Public) 358  
set\_memory\_allocation\_increment() (Public) 119, 374, 384, 440, 597, 599, 601, 603, 605, 607, 609, 611, 613, 615, 619, 621, 623, 625, 627, 631, 633, 635, 639, 644, 647, 651  
set\_memory\_allocation\_increment() (Public, static) 119, 374, 440, 597, 599, 601, 603, 605, 607, 609, 611, 613, 615, 619, 621, 623, 625, 627, 631, 633, 635, 639, 644, 647, 651  
set\_mesh() (Public) 498  
set\_name() (Public) 71  
set\_name() (Public, static) 71  
set\_neighbor() (Public) 526  
set\_neighbor\_points() (Private) 740

# Function Index

set\_neighborhood\_bounds() (Public) 325  
set\_neighbors\_drawn() (Private) 529  
set\_next\_face\_plane() (Private) 695  
set\_no\_corners() (Public) 374  
set\_nodes\_perm() (Private) 585  
set\_nodes\_permanent() (Private) 112  
set\_num\_hexes() (Public) 502  
set\_num\_levels() (Public) 502  
set\_objective() (Public) 358, 375  
set\_opp\_dir() (Public) 697  
set\_opp\_edge() (Private) 694  
set\_option() (Public) 71  
set\_option() (Public, static) 71  
set\_owners() (Private) 207  
set\_part\_number() (Public) 254  
set\_periodic() (Public) 639  
set\_perm\_and\_bdy() (Public) 162  
set\_picture\_segment() (Public) 224, 235  
set\_proj() (Protected) 678  
set\_proper\_mesh\_edge\_owners() (Public)  
    269  
set\_refface\_mesh\_scheme() (Private, static)  
    284  
set\_rhs() (Public) 358  
set\_screen\_coords() (Public) 744  
set\_singlet\_orientation() (Private) 754  
set\_size() (Public) 245  
set\_size\_smallest\_edge() (Public) 545  
set\_sorted\_flag() (Protected) 512  
set\_sum\_odd() (Public) 374  
set\_third\_node() (Private) 694  
set\_use\_count\_to\_zero() (Public) 473  
set\_variable\_name() (Public) 358  
set\_vertex\_type() (Public) 489, 593  
set\_weaver\_window() (Public, static) 757  
setConstantSize() (Public) 489  
setCurvatureFactor() (Public) 489  
setting\_handler() (Private, static) 451  
setting\_handler() (Public, static) 403  
setup\_associativity\_nodesets() (Private)  
    300  
setup\_element\_blocks() (Private) 287, 300,  
    765  
setup\_hoops\_and\_x() (Protected) 233  
setup\_nodesets() (Private) 300  
setup\_segments() (Private) 407  
setup\_xslicing\_plane() (Private) 407  
setup\_yslicing\_plane() (Private) 407  
setup\_zslicing\_plane() (Private) 408  
seven\_neighbor\_improvement() (Private)  
    106  
share\_2cell() (Private) 535, 557  
share\_2cell() (Public) 536, 558  
share\_face() (Public) 728  
shared\_edge() (Public) 140, 164  
shared\_edges() (Public) 140  
shared\_face() (Public) 133, 164, 290  
shared\_face() (Public, virtual) 144  
shared\_face\_and\_hexes() (Public) 710  
shared\_node() (Public) 133, 710  
shared\_sheet() (Public) 724  
shared\_vertex() (Public) 279  
sheet\_1() (Public) 714  
sheet\_2() (Public) 714  
sheet\_chord() (Public) 527, 535, 558, 715  
sheet\_chord\_1() (Public) 714  
sheet\_chord\_2() (Public) 714  
sheet\_chord\_at\_level() (Public) 744  
sheet\_chord\_list() (Public) 742  
sheet\_chord\_points() (Public) 521  
sheet\_chord\_sheet() (Public) 715  
sheet\_color() (Private) 232  
SheetChord (Public, constructor) 519  
SheetChord (Public, destructor) 519  
sheetchord\_angle() (Public) 741  
sheetchord\_angle\_given\_size() (Public,  
    static) 741  
SheetChordPoint (Public, constructor) 525  
SheetChordPoint (Public, destructor) 525  
SheetEdge (Public, constructor) 535  
sheets() (Public) 397  
shift() (Public) 245  
shift\_edge() (Private) 576  
should\_shrink() (Private) 207  
should\_write() (Private) 409  
show\_all\_object\_memory() (Public, static)  
    384  
show\_faces\_after\_node\_merge() (Public)  
    333  
show\_faces\_before\_node\_merge() (Public)  
    333  
show\_faces\_during\_node\_merge() (Public)

# Function Index

333  
show\_memory() (Public) 225, 241  
show\_object\_memory() (Public, static) 384  
shrink() (Public) 63, 200  
shrunk\_node() (Public) 211, 431, 627  
side\_chord() (Public) 730  
side\_handler() (Public, static) 283  
side\_skew\_left() (Private) 109  
side\_skew\_right() (Private) 109  
SideSet (Private, constructor) 542  
SideSet (Public, constructor) 541  
SideSet (Public, destructor) 541  
sideset\_color() (Private) 232  
sideset\_list() (Public) 299  
simple\_sub\_map() (Public) 571  
simple\_vol\_map() (Public) 688  
single\_skew\_left() (Private) 110  
single\_skew\_right() (Private) 110  
six\_neighbor\_improvement() (Private) 106  
size() (Public) 63, 267, 419  
size\_at\_point() (Public) 487  
size\_handler() (Public, static) 283  
sizing\_data\_ok() (Private) 561  
sizing\_distance\_at() (Public) 468  
sizing\_func\_type() (Public) 268  
sizing\_function() (Public) 487  
sizing\_ref\_face() (Public) 561, 635  
SizingTool (Public, constructor) 545  
SizingTool (Public, destructor) 545  
skew\_seam() (Private) 425  
skip\_joins() (Public) 755  
SLLList (Public, constructor) 548  
SLLList (Public, virtual, destructor) 548  
SLLListItem (Private, constructor) 547  
SLLListIterator (Public, constructor) 549  
smooth() (Private) 207  
smooth() (Public) 434, 490, 495, 505, 520,  
    526, 742  
smooth() (Public, static) 521  
smooth() (Public, virtual) 474  
smooth\_boundary() (Public) 346  
smooth\_entity\_collection() (Public) 581  
smooth\_flag() (Public) 525  
smooth\_handler() (Public, static) 389  
smooth\_hex\_location() (Public) 730  
smooth\_hexes() (Public) 717  
smooth\_hexes\_along\_chord() (Public) 717  
smooth\_interior\_nodes() (Public) 346, 581  
smooth\_layer() (Protected) 677  
smooth\_node() (Private) 561  
smooth\_node\_list() (Public) 346  
smooth\_node\_merge() (Private) 332  
smooth\_nodes() (Private) 561  
smooth\_scheme() (Public) 473  
smooth\_scheme\_name() (Public) 473  
smooth\_sheets() (Public) 332  
smooth\_target\_face() (Protected) 677  
smoothing\_done() (Public) 730  
soft\_edge\_bounds() (Public) 374  
solution() (Public) 358  
solve() (Public) 358  
solve\_interval\_assignment() (Private)  
    575  
sort\_angle() (Private) 79  
sort\_angles() (Public) 571  
sort\_list() (Protected) 512  
sort\_source\_entities() (Private) 679  
source\_cubitedge() (Public) 485  
source\_cubitnode() (Public) 485  
source\_face() (Public) 502  
source\_face() (Public, virtual) 472  
source\_face\_list() (Public) 502  
source\_refedge() (Public) 485  
source\_refface() (Public) 485  
source\_refvertex() (Public) 485  
source\_target\_mesh\_compatible() (Protected)  
    677  
spacing\_list() (Public) 94, 96  
special\_vertex\_setting() (Private) 595  
sphere() (Public) 310  
splice\_sheets() (Public) 743  
splice\_two\_to\_one() (Private) 434, 740  
split\_and\_combine() (Private) 111  
split\_at\_node() (Private) 427  
split\_boundary\_invalid() (Private) 112  
split\_by\_rotate\_left() (Private) 112  
split\_by\_rotate\_right() (Private) 112  
split\_chord() (Public) 716, 730  
split\_complete() (Private) 713  
split\_cut\_bef\_knife() (Private) 714  
split\_in\_corner() (Private) 112  
split\_incomplete() (Private) 713

# Function Index

split\_insert\_two\_faces() (Private) 108  
split\_into\_components() (Private) 433  
split\_loop() (Private) 427  
split\_loop() (Public) 724, 743  
split\_owner() (Public, virtual) 67, 71, 73  
spot\_7th\_node() (Private) 330, 349  
spot\_7th\_node() (Public, virtual) 86  
spot\_8th\_node() (Private) 330, 349  
spot\_8th\_node() (Public, virtual) 86  
spotlight\_key() (Public) 221, 238  
squeeze\_double\_row() (Private) 106  
squeeze\_face() (Private) 107  
squeeze\_invalid\_face() (Private) 111  
stair\_block() (Public) 462  
stair\_interval() (Public) 462  
StairTool (Public, constructor) 552  
StairTool (Public, destructor) 552  
StarNode (Public, constructor) 555  
starred() (Public) 211, 431, 627  
start() (Public) 392  
start\_face\_sheet() (Public) 724  
start\_node() (Public) 133, 251  
start\_on\_corner() (Private) 424  
start\_on\_reversal() (Private) 425  
start\_on\_side() (Private) 424  
start\_up\_screen() (Public) 682  
starting\_points() (Public) 520  
startRefVertex() (Public) 467  
state() (Public) 438  
state\_remove() (Public) 757  
state\_unknown() (Public) 439  
STC\_buffer() (Public) 756  
STC\_fixups() (Public) 756  
StcEdge (Public, constructor) 557  
std\_dev() (Public) 455  
step() (Public) 199, 458  
step\_and\_get\_item() (Protected) 201  
step\_around\_node() (Public) 759  
stored\_temp() (Public) 351  
straight\_nodes() (Private) 695  
Strider (Public, constructor) 561  
Strider (Public, destructor) 561  
string\_to\_color\_id() (Private, static) 321  
string\_to\_enum() (Public, static) 254  
string\_to\_mesh\_enum() (Protected, static)  
471  
string\_to\_smooth\_enum() (Protected, static)  
471  
sub\_face() (Public) 503  
sub\_map() (Public) 570  
sub\_map\_volume() (Public) 696  
sub\_matrix() (Public) 154  
SubDomain (Public, constructor) 563  
SubDomain (Public, destructor) 563  
submap\_curve\_type() (Private) 373  
submap\_side\_edge\_list() (Private) 373  
submap\_smooth() (Public) 490  
SubMapNode (Public, constructor) 565  
SubMapNode (Public, destructor) 565  
SubMapTool (Public, constructor) 570  
SubMapTool (Public, destructor) 570  
substitute\_chord() (Public) 729  
substitute\_chord1() (Private) 727  
substitute\_chord2() (Private) 727  
substitute\_chord3() (Private) 728  
substitute\_neighbor() (Public) 526  
substitute\_sheet\_chord() (Public) 716  
substr() (Public) 184  
subtract() (Public) 311  
sum\_even\_constraint() (Public) 375  
summarize\_data() (Protected) 453  
sup\_face() (Public) 488  
super\_sizing\_function() (Private) 484  
superdrive\_on\_path() (Private) 736  
surface() (Private) 423  
surface\_color() (Private) 232  
surface\_face() (Public) 745  
surface\_type() (Public) 488  
SurfDistributeTool (Public, constructor) 573  
SurfDistributeTool (Public, destructor) 573  
SurfMapTool (Public, constructor) 577  
SurfMapTool (Public, destructor) 577  
SurfMeshTool (Public, constructor) 580  
SurfMeshTool (Public, destructor) 580  
SurfMorphTool (Public, constructor) 585  
SurfMorphTool (Public, destructor) 585  
SurfSubDomain (Public, constructor) 587  
SurfSubDomain (Public, destructor) 587  
SurfVertexType (Public, constructor) 593  
SurfVertexType (Public, destructor) 593  
surround\_2cells() (Public) 711  
surround\_3cell() (Public) 711

# Function Index

surrounding\_voxels() (Public) 192, 706  
swap() (Public) 706  
swap\_face\_use() (Public) 289  
swap\_sheet\_chords() (Public) 714  
sweep\_FACE\_about\_axis() (Public) 57  
sweep\_FACE\_along\_vector() (Public) 57  
sweep\_get\_FACES\_of\_RefFaces() (Private) 315  
sweep\_get\_refface\_list() (Private, static) 304  
sweep\_handler() (Public, static) 303  
sweep\_reffaces() (Private, static) 304  
sweep\_rotational() (Private, static) 304  
sweep\_rotational() (Public) 310  
sweep\_translational() (Private, static) 304  
sweep\_translational() (Public) 310  
SweepTool (Public, constructor) 595  
SweepTool (Public, destructor) 595  
switch\_diagonal\_ccw() (Private) 107  
switch\_diagonal\_cw() (Private) 107  
switch\_nodes() (Public) 412  
switch\_parent() (Public) 593  
switch\_sheet\_chords() (Public) 716  
switch\_to\_closure() (Public) 334

## T

---

take\_out() (Private) 365  
take\_out\_non\_split() (Private) 365  
take\_tuck() (Private) 425  
tangent() (Public) 467  
target\_cubitedge() (Public) 485  
target\_cubitnode() (Public) 485  
target\_face() (Public) 502  
target\_face() (Public, virtual) 472  
target\_refedge() (Public) 485  
target\_refvertex() (Public) 485  
TDCellIndex (Public, constructor) 597  
TDCellIndex (Public, virtual, destructor) 597  
TDCenterNode (Public, constructor) 599  
TDCopied (Public, constructor) 601  
TDDistance (Public, constructor) 603  
TDDistance (Public, virtual, destructor) 603  
TDGenesisID (Public, constructor) 605  
TDGenesisID (Public, virtual, destructor) 605  
TDGeometrySizing (Public, constructor) 607

TDGeometrySizing (Public, destructor) 607  
TDHexKnowing (Public, constructor) 609  
TDLaplace (Public, constructor) 611  
TDLaplace (Public, destructor) 611  
TDLayer (Public, constructor) 613  
TDLPEdge (Public, constructor) 615  
TDLPEdge (Public, destructor) 615  
TDMorph (Public, constructor) 619  
TDMorph (Public, destructor) 619  
TDNodeHardLine (Public, constructor) 621  
TDNodeHardLine (Public, destructor) 621  
TDNormal (Public, constructor) 623  
TDNormal (Public, virtual, destructor) 623  
TDPaver (Public, constructor) 625  
TDPaver (Public, destructor) 625  
TDPillow (Public, constructor) 627  
TDPillow (Public, destructor) 627  
TDProjection (Public, constructor) 631  
TDSizing (Public, constructor) 633  
TDSizing (Public, virtual, destructor) 633  
TDStrider (Public, constructor) 635  
TDStrider (Public, destructor) 635  
TDSubMap (Public, constructor) 637  
TDSubMap (Public, destructor) 637  
TDTipton (Public, constructor) 643  
TDTipton (Public, destructor) 643  
TDUVSpace (Public, constructor) 647  
TDWeight (Public, constructor) 651  
temp\_color() (Private) 232  
temp\_entry() (Public) 615, 616  
temp\_location() (Public) 643  
temp\_x() (Public) 643  
temp\_y() (Public) 643  
temp\_z() (Public) 643  
test() (Public) 60  
test\_h1\_copies() (Private) 426  
test\_neighbors() (Public) 521, 716  
test\_sizing\_function() (Private) 484  
test\_suite() (Public) 366  
test\_upgrade\_to\_edge() (Private) 207  
test\_upgrade\_to\_surface() (Private) 207  
test\_upgrade\_to\_vertex() (Private) 207  
text\_color() (Private) 232  
text\_size() (Public) 223, 241  
third\_cell() (Public) 759  
third\_last\_point() (Public) 520

# Function Index

three\_neighbor\_improvement() (Private) 106  
three\_sided\_rectangle\_constraints() (Public) 375  
three\_to\_one() (Private) 106  
three\_triple\_five() (Private) 106  
times\_is\_neighbor() (Public) 527  
TimeVal (Protected, constructor) 653  
TimeVal (Public, constructor) 653  
tipton\_smooth() (Private) 579  
tipton\_smooth() (Public) 346  
title\_string() (Public, static) 295  
Tool (Public, constructor) 657  
Tool (Public, destructor) 657  
tool\_data() (Private) 663  
ToolData (Protected, constructor) 659  
ToolData (Public, virtual, destructor) 659  
ToolDataUser (Public, constructor) 663  
ToolDataUser (Public, destructor) 663  
top\_weave\_level() (Public) 755  
torus() (Public) 310  
total\_depth() (Public) 94  
total\_nodes\_in\_grid\_cells() (Public) 324  
trade\_sheet\_chords() (Public) 521  
transfer\_chord() (Public) 716  
transfer\_points\_forward() (Public) 520  
transfer\_points\_reverse() (Public) 521  
transform() (Private) 740  
transform() (Public) 526  
transform() (Public, static) 521  
transform\_body() (Private) 315  
transform\_mesh() (Public) 312  
transform\_node() (Private) 315  
transform\_point() (Private) 233  
transform\_ref\_edges\_mesh() (Public) 312  
transform\_ref\_vertices\_mesh() (Public) 312  
transform\_ref\_volumes\_mesh() (Public) 313  
transform\_refFaces\_mesh() (Public) 312  
TranslateTool (Public, constructor) 665  
TranslateTool (Public, destructor) 665  
transpose() (Public) 153  
traversal() (Public) 571  
traverse() (Private) 373  
Tree (Public, constructor) 667  
tree\_handler() (Public, static) 403  
triangle\_at\_boundary() (Private) 110  
triangle\_constraints() (Public) 375  
triangleFormation() (Public) 86  
triangleFormationAngleDegrees() (Public) 490  
triangleFormationAngleRadians() (Public) 490  
TriangleTool (Public, constructor) 671  
TriangleTool (Public, destructor) 671  
TriElementTool (Public, constructor) 673  
TriElementTool (Public, destructor) 673  
try\_shift() (Private) 576  
try\_side() (Private, static) 148  
tuck\_and\_wedge() (Private) 425  
turn\_angle\_metric() (Protected) 377  
turn\_ccw\_corner() (Private) 688  
turn\_cw\_corner() (Private) 687  
turn\_off() (Public) 593  
turn\_off\_vertex() (Public) 639  
turn\_on() (Public) 593  
turn\_vertex\_off() (Public) 489  
twin\_chord() (Public) 714  
twin\_node() (Public) 619  
twin\_sheet\_chord() (Public) 522, 716  
twin\_sheet\_edge() (Public) 535, 558  
two\_cell\_hexes() (Public) 744  
two\_cell\_points() (Public) 744  
two\_d\_cross\_another() (Private) 426  
two\_d\_cross\_self() (Private) 426  
two\_middle\_nodes() (Private) 108  
two\_neighbor\_improvement() (Private) 106  
two\_row\_transition() (Private) 109  
two\_valent\_hardpoint() (Private) 106  
two\_valent\_hp\_check() (Private) 106  
TwoHalfDimTool (Public, constructor) 675  
TwoHalfDimTool (Public, virtual, destructor) 675  
type() (Private, static) 202, 246, 512  
type() (Public) 289, 411, 581  
type() (Public, virtual) 141, 144

## U

---

u\_v\_coordinates() (Public) 489

# Function Index

u\_v\_params() (Public) 488  
unassociate() (Public) 351  
unassociate\_from\_LP() (Public) 374  
unbreak\_neighbors() (Private) 434  
unbreak\_neighbors() (Public) 745  
uncross\_first() (Public, static) 756  
underlying\_edge() (Public) 96  
underlying\_LOOP() (Public) 96  
unget\_input() (Public) 681  
unhook\_BODY\_from\_refentities() (Public)  
    56  
unite() (Public) 311, 312  
unload\_sizing\_data() (Public) 484  
unmark() (Private) 433  
unmark\_faces() (Private) 688  
unmark\_global\_edges() (Public, static) 134  
unmark\_global\_faces() (Public, static) 141  
unmark\_global\_nodes() (Public, static) 162  
unmark\_nodes() (Private) 688  
unmark\_points() (Private) 736  
untreated\_sideset\_faces() (Private) 301  
up() (Public) 219, 236, 237  
update() (Public) 222, 239  
update\_boundary\_nodes() (Public) 86  
update\_hack() (Public, static) 321  
update\_join\_neighbors() (Public) 758  
update\_lighting() (Private) 232  
update\_sheet\_chords() (Public) 728  
update\_state\_lists() (Public) 745  
update\_states() (Private) 331, 349  
update\_states() (Public, virtual) 86  
update\_viewing() (Public) 222, 239  
update\_voidface\_untried\_proto\_states()  
    ) (Private) 349  
usage() (Public) 320  
use\_count() (Public) 473  
use\_multiple\_windows() (Public, static) 757  
use\_superdrive() (Public, static) 756  
UserInterface (Protected, constructor) 682  
UserInterface (Public, destructor) 681

## V

---

v\_to\_x() (Private) 108  
valid\_face() (Private) 111  
valid\_node\_merge() (Public) 86

validate\_handler() (Public, static) 389  
validate\_sheets() (Public, static) 758  
value() (Public) 481  
variable\_is\_integer() (Public) 358  
vector\_handler() (Public, static) 321  
vector\_size() (Public) 223, 241  
vector\_style() (Public) 223, 241  
verify\_no\_TDProj() (Protected) 678  
verify\_structure() (Public) 435  
version() (Public, static) 403  
vertex\_color() (Private) 232  
video\_handler() (Public, static) 321  
video\_init() (Public) 223  
video\_on() (Public) 223  
video\_snap() (Public) 223  
viewport() (Public) 221  
void\_boundary() (Public) 503  
vol\_map\_mark() (Public) 141, 163  
VolMapTool (Public, constructor) 688  
VolMapTool (Public, destructor) 688  
VolMeshTool (Public, constructor) 691  
VolMeshTool (Public, destructor) 691  
VolSubMapTool (Public, constructor) 696  
VolSubMapTool (Public, destructor) 696  
volume() (Public) 59  
volume\_cells() (Public) 755  
volume\_sheets() (Public) 755  
VolumeEdgeType (Public, constructor) 699  
VolumeEdgeType (Public, destructor) 699  
VolVoidBoundary (Public, constructor) 701  
VolVoidBoundary (Public, destructor) 701  
VolVoidSepTool (Public, constructor) 703  
VolVoidSepTool (Public, destructor) 703  
voxel\_centroid() (Public) 192  
voxel\_index() (Public) 192, 706  
voxel\_location() (Public) 192  
voxel\_size\_x() (Public) 705  
voxel\_size\_y() (Public) 705  
voxel\_size\_z() (Public) 705  
voxel\_tolerance() (Public) 706  
VoxelTool (Public, constructor) 705  
VoxelTool (Public, destructor) 705

## W

---

warning\_flag() (Public) 156

# Function Index

weave() (Public) 743, 756  
weave\_hex() (Public) 729  
weave\_new\_hex() (Public, static) 729  
weave\_old() (Public) 743  
weave\_state() (Public) 519  
web\_node() (Public) 640  
webcut\_BODY() (Public) 57  
webcut\_cleanup() (Private, static) 303  
webcut\_failed() (Private, static) 303  
webcut\_generate\_acis\_cutting\_tool()  
    (Private, static) 304  
webcut\_generate\_acis\_plane\_cutting\_to  
    ol() (Private, static) 304  
webcut\_get\_refbody\_list() (Private, static)  
    303  
webcut\_handler() (Public, static) 303  
webcut\_imprint() (Public) 310  
webcut\_merge() (Private, static) 304  
wedge\_contains() (Private) 206  
weight() (Public) 651  
weight\_function\_1() (Public) 366  
weight\_function\_2() (Public) 366  
weight\_function\_3() (Public) 366  
weight\_function\_4() (Public) 366  
weight\_function\_5() (Public) 366  
weight\_scheme() (Public) 141, 473  
whisker\_cells() (Public) 557, 724, 730  
whisker\_chord() (Public) 519  
whisker\_chords() (Public) 760  
whisker\_face() (Public) 527  
whisker\_face() (Public, static) 724  
whisker\_hex() (Public) 525  
whisker\_hex() (Public, static) 87  
whisker\_knife\_handler() (Public, static)  
    389  
whisker\_sheet() (Public) 519, 526, 709  
whisker\_sheets() (Public) 760  
whisker\_weaver() (Public) 503, 741  
whisker\_weaving\_active() (Public, static)  
    756  
WhiskerCell (Public, constructor) 709  
WhiskerCell (Public, destructor) 709  
WhiskerChord (Public, constructor) 714  
WhiskerChord (Public, destructor) 714  
WhiskerFace (Public, constructor) 723  
WhiskerFace (Public, destructor) 723  
WhiskerHex (Public, constructor) 727  
WhiskerHex (Public, destructor) 727  
WhiskerKnife (Public, constructor) 736  
WhiskerKnife (Public, destructor) 736  
WhiskerSheet (Public, constructor) 741  
WhiskerSheet (Public, virtual, destructor) 741  
WhiskerWeaver (Public, constructor) 755  
WhiskerWeaver (Public, virtual, destructor)  
    755  
window\_id() (Public) 216, 234, 235  
window\_size() (Public) 221, 238  
wrapup\_fastq() (Public, static) 283  
write() (Public) 253  
write() (Public, virtual) 297  
write\_coordinate\_names() (Private) 300  
write\_coordinates() (Private) 287, 300, 765  
write\_element\_blocks() (Private) 287, 300,  
    765  
write\_element\_order\_map() (Private) 300  
write\_file() (Public) 287, 299, 409, 765  
write\_fred\_triangles() (Public) 253  
write\_header() (Private) 287, 300, 765  
write\_materials() (Private) 287, 765  
write\_nodesets() (Private) 300  
write\_point() (Private) 409  
write\_properties() (Private) 300  
write\_qa\_records() (Private) 300  
write\_quality() (Private) 300  
write\_sheet() (Private) 409  
write\_sheets() (Private) 409  
write\_sidesets() (Private) 300  
write\_surface\_sidesets() (Private) 300  
write\_volume\_sidesets() (Private) 300  
write\_xpatch\_triangles() (Public) 253

---

**X**

---

x\_range() (Public) 125  
XpatchTool (Protected, constructor) 765  
XpatchTool (Public, destructor) 765

---

**Y**

---

y\_range() (Public) 125

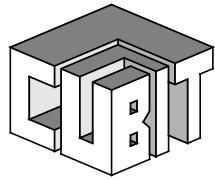
# Function Index

## Z

---

`z_range()` (Public) 125  
`zero_search()` (Protected) 517  
`zero_temp()` (Public) 351  
`zoom_camera()` (Private) 332  
`zoom_cursor()` (Public) 220, 237  
`zoom_entity()` (Public) 220, 237  
`zoom_handler()` (Public, static) 321  
`zoom_parameters()` (Public) 219, 220, 237  
`zoom_screen()` (Public) 220, 237





---

# Constants and Defined Macros Index

## A

---

ACIS\_GEOMETRY\_ENGINE 62  
ANIMATION\_READ 338  
ANIMATION\_WRITE 338  
ARRAYBASEDCONTAINER\_HPP 65  
ATTRIB\_CUBIT\_OWNER\_CLASS 67  
ATTRIB\_CUBIT\_OWNER\_LEVEL 67  
ATTRIB\_GTC\_CLASS 69  
ATTRIB\_GTC\_LEVEL 69  
ATTRIB\_GTC\_NAME\_CLASS 71  
ATTRIB\_GTC\_NAME\_LEVEL 71  
ATTRIB\_PARENTS\_CLASS 73  
ATTRIB\_PARENTS\_LEVEL 73  
ATTRIB\_SNL\_CLASS 75  
ATTRIB\_SNL\_LEVEL 75  
ATTRIBUTECOMMANDS\_HPP 78  
AUTOVERTEXTYPE\_HPP 80

## B

---

BASEHEXER\_HPP 88  
BLADE\_EDGES\_HPP 90  
BLADE\_EDGES\_TREE\_HPP 91  
BORDER 752  
BOUNDARYLAYER\_HPP 94  
BOUNDARYLAYEREDGE\_HPP 97  
BOUNDARYLAYERFACE\_HPP 100  
BOUNDARYLAYERTOOL\_HPP 103

## C

---

CLEANUP\_HPP 113  
COLLINEAR 135  
COMMANDHANDLER\_HPP 117  
CommonDefine(typePtr, notSorted) 204,  
514  
CommonSortedDefine(name, typePtr) 514  
CONTROL\_POINT\_HPP 120

COPIEDDATA\_HPP 121  
CPU\_TIMER 123  
CUBIT\_COLLECTION\_HPP 127  
CUBIT\_CONTAINER\_HPP 129  
CUBIT\_DEBUG\_1 158  
CUBIT\_DEBUG\_10 158  
CUBIT\_DEBUG\_11 158  
CUBIT\_DEBUG\_12 158  
CUBIT\_DEBUG\_13 158  
CUBIT\_DEBUG\_14 158  
CUBIT\_DEBUG\_15 158  
CUBIT\_DEBUG\_16 158  
CUBIT\_DEBUG\_17 158  
CUBIT\_DEBUG\_18 158  
CUBIT\_DEBUG\_19 158  
CUBIT\_DEBUG\_2 158  
CUBIT\_DEBUG\_20 158  
CUBIT\_DEBUG\_21 158  
CUBIT\_DEBUG\_22 158  
CUBIT\_DEBUG\_23 158  
CUBIT\_DEBUG\_24 158  
CUBIT\_DEBUG\_25 158  
CUBIT\_DEBUG\_26 158  
CUBIT\_DEBUG\_27 158  
CUBIT\_DEBUG\_28 158  
CUBIT\_DEBUG\_29 158  
CUBIT\_DEBUG\_3 158  
CUBIT\_DEBUG\_30 158  
CUBIT\_DEBUG\_31 158  
CUBIT\_DEBUG\_32 158  
CUBIT\_DEBUG\_33 158  
CUBIT\_DEBUG\_34 158  
CUBIT\_DEBUG\_35 158  
CUBIT\_DEBUG\_36 158  
CUBIT\_DEBUG\_37 158  
CUBIT\_DEBUG\_38 158  
CUBIT\_DEBUG\_39 158  
CUBIT\_DEBUG\_4 158

# Constants and Defined Macros Index

CUBIT_DEBUG_40	158	CUBIT_SHEET_HPP	180
CUBIT_DEBUG_41	158	CUBIT_STACK_HPP	181
CUBIT_DEBUG_42	158	CUBIT_WARNING	158
CUBIT_DEBUG_43	158	CUBITBOX_HPP	126
CUBIT_DEBUG_44	158	CUBITENTITY_HPP	138
CUBIT_DEBUG_45	158	CUBITMATRIX_HPP	154
CUBIT_DEBUG_46	158	CUBITMESSAGE_HPP	158
CUBIT_DEBUG_47	158	CUBITOBJECT_HPP	132
CUBIT_DEBUG_48	159	CUBITPLANE_HPP	173
CUBIT_DEBUG_49	159	CUBITVECTOR_HPP	190
CUBIT_DEBUG_5	158	CUBITVOXEL_HPP	193
CUBIT_DEBUG_50	159	CURVESUBDOMAIN_HPP	196
CUBIT_DEBUG_51	159		
CUBIT_DEBUG_52	159		
CUBIT_DEBUG_53	159		
CUBIT_DEBUG_54	159		
CUBIT_DEBUG_55	159		
CUBIT_DEBUG_56	159		
CUBIT_DEBUG_57	159		
CUBIT_DEBUG_58	159		
CUBIT_DEBUG_59	159		
CUBIT_DEBUG_6	158		
CUBIT_DEBUG_60	159		
CUBIT_DEBUG_7	158		
CUBIT_DEBUG_8	158		
CUBIT_DEBUG_9	158		
CUBIT_DIAGNOSTIC	158		
CUBIT_EDGE_HPP	135		
CUBIT_ERROR	158		
CUBIT_FACE_HPP	142		
CUBIT_FULLHEX_HPP	291		
CUBIT_HEX_HPP	145		
CUBIT_INFO	158		
CUBIT_KNIFE_HPP	150		
CUBIT_LOGICAL_HPP	151		
CUBIT_MAX(a,b)	132		
CUBIT_MAX_4(a,b,c,d)	132		
CUBIT_MIN(a,b)	132		
CUBIT_MIN_4(a,b,c,d)	132		
CUBIT_NODE_ARRAY	168		
CUBIT_NODE_ARRAY_HPP	170		
CUBIT_NODE_ARRAY1D_HPP	168		
CUBIT_NODE_ARRAY2D_HPP	169		
CUBIT_NODE_HEX_HPP	413		
CUBIT_NODE_HPP	166		
CUBIT_PTR_ARRAY_HPP	175		
		CUBIT_SHEET_HPP	180
		CUBIT_STACK_HPP	181
		CUBIT_WARNING	158
		CUBITBOX_HPP	126
		CUBITENTITY_HPP	138
		CUBITMATRIX_HPP	154
		CUBITMESSAGE_HPP	158
		CUBITOBJECT_HPP	132
		CUBITPLANE_HPP	173
		CUBITVECTOR_HPP	190
		CUBITVOXEL_HPP	193
		CURVESUBDOMAIN_HPP	196
		<b>D</b>	
		DEBUG_FLAG	159
		DEGREES_TO_RADIANS(angle)	132
		DIAGNOSTIC	159
		DIAGNOSTIC_FLAG	159
		DLLIST_HPP	204
		DLListdeclare(name, typePtr)	204
		DO_INTERSECT	135
		DONT_INTERSECT	135
		DOUBLET_PILLOWER_HPP	209
		DOUBLET_PILLOWER_TD_HPP	213
		DRAWINGTOOL_HPP	226
		DRAWINGTOOLINSTANCE_HPP	243
		DYNAMICARRAY_HPP	246
		DynamicArrayDeclare(name, typePtr)	246
		DYNLOOPLIST_HPP	247
		<b>E</b>	
		EDGE_USE_HPP	252
		EDGEMESHTOOL_HPP	250
		ELEMENT_QUALITY_HPP	264
		ELEMENTBLOCK_HPP	255
		ELEMENTBLOCK1D_HPP	258
		ELEMENTBLOCK2D_HPP	260
		ELEMENTBLOCK3D_HPP	261
		EXODUSMESH_HPP	269
		<b>F</b>	
		FACE_TYPES_HPP	271
		FACE_USE_HPP	275
		FALSE	132

# Constants and Defined Macros Index

FASTQCLASSES\_HPP 281  
FASTQCOMMANDS\_HPP 284  
FILECOMMANDS\_HPP 285  
FREDTOOL\_HPP 288

## G

---

GENESIS\_ENTITY\_HPP 298  
GENESISCOMMANDS\_HPP 295  
GENESISSTOOL\_HPP 302  
GEOMETRY\_SIZING\_TOOL\_HPP 308  
GEOMETRYCOMMANDS\_HPP 305  
GEOMETRYTOOL\_HPP 317  
GETLONGOPT\_HPP 320  
GRAPHICSCOMMANDS\_HPP 322  
GRID\_SEARCH\_HPP 328

## H

---

HEX\_KNOWING\_NODE\_HPP 339  
HEX\_QUALITY\_CONTROLLER\_HPP 343  
HEX\_QUALITY\_HPP 342  
HEX\_TO\_VOID\_HPP 350  
HEXER\_HPP 338  
HEXTOOL\_HPP 346

## I

---

INFO\_FLAG 159  
INTERVAL\_LINEAR\_PROGRAM\_HPP 355

## L

---

LINEAR\_PROGRAM\_HPP 361  
LOOP\_COLLAPSER\_HPP 367  
LOOP\_HPP 363  
LOOP\_INTERVAL\_HPP 369  
LOOP\_JOINER\_HPP 372

## M

---

MAP\_TOOL\_SUPPORT\_HPP 378  
MAPPING\_FACE\_HPP 376  
MAX\_INSIDE\_LOOPS 752  
MAX\_INSTANCES 226  
MAX\_WEAVE\_LEVELS 763  
MEMORY\_ALLOCATION\_HPP 380

MEMORY\_BLOCK\_HPP 382  
MEMORY\_MANAGER\_HPP 385  
MemoryAllocationCode(ClassName) 380  
MemoryAllocationFunctions() 380  
MemoryAllocationParameters(ClassName)  
380  
MESH\_INTERSECT\_HPP 393  
MESHENITIY\_HPP 388  
MESHINGCOMMANDS\_HPP 389  
MESHTOOL\_HPP 395  
MODEL\_HPP 402  
MODEL\_VIEWER\_HPP 406  
MODELCOMMANDS\_HPP 403  
MOUSE\_HPP 408  
MUSEOUTPUT\_HPP 409

## N

---

NODESET\_HPP 417  
NULL 132  
NUM\_ITERATIONS 573

## O

---

OPPOSITE\_DIRECTION 534

## P

---

PARALLELMESHTOOL\_HPP 420  
PARENTS\_CLASS 421  
PAVER\_HPP 430  
PILLOW\_NODE\_HPP 432  
PILLOW\_SHEET\_HPP 435  
POINT\_INSIDE 193  
POINT\_ON 193  
POINT\_OUTSIDE 193  
PRINT\_DEBUG 159  
PRINT\_ERROR 159  
PRINT\_INFO 159  
PRINT\_WARNING 159  
PROTO\_HEX\_HPP 441  
PYRAMID\_HPP 443  
PYRAMIDTOOL\_HPP 446

## Q

---

QUAD\_QUALITY\_CONTROLLER\_HPP 449

# Constants and Defined Macros Index

QUAD\_QUALITY\_HPP 448  
QUALITY\_CONTROLLER\_HPP 454  
QUALITY\_SUMMARY 455  
QUALITYCOMMANDS\_HPP 451  
QUEUE\_HPP 459  
Queuedeclare(name, typePtr) 459

## R

---

RANGE 132  
REFBODY\_HPP 464  
REFEDGE\_HPP 470  
REFENTITY\_HPP 477  
REFENTITYNAMEMAP\_HPP 482  
REFFACE\_HPP 494  
REFGROUP\_HPP 496  
REFVERTEX\_HPP 499  
REFVOLUME\_HPP 507  
ROTATETOOL\_HPP 510

## S

---

SAME\_SIGNS( a, b ) 193  
SDLLIST\_HPP 514  
SDLListdeclare(name, typePtr, functionName, functionType) 514  
SELF\_CROSSING\_HPP 516  
SELF\_CROSSING\_LOOP\_HPP 518  
SEPERATION 752  
SetDynamicMemoryAllocation(memManager) 385  
SHEET\_CHORD\_HPP 524  
SHEET\_CHORD\_POINT\_HPP 534  
SHEET\_EDGE\_HPP 536  
SHEET\_EDGE\_TREE\_HPP 537  
SHEET\_EDGE\_TREE\_QUEUE\_HPP 539  
SIDESSET\_HPP 543  
sign(x) 524, 721  
SIZING\_TOOL\_HPP 545  
SLLIST\_HPP 547  
SLListdeclare(name, typePtr) 549  
SLListIteratordeclare(name, typePtr) 549  
SORTED\_NODE\_DATA\_ARRAY\_INCREMENT 338  
SSDLListIntrinsicdeclare(name, typeP-

tr) 514  
Stackdeclare(name, typePtr) 181  
STAR\_NODE\_HPP 556  
STC\_EDGE\_HPP 559  
STRIDER\_HPP 562  
STRING\_HPP 184  
SUBDOMAIN\_HPP 564  
SUBMAPNODE\_HPP 567  
SUBMAPTOOL\_HPP 572  
SURF\_MAPTOOL\_HPP 578  
SURFDISTRIBUTETOOL\_HPP 573  
SURFMESHTOOL\_HPP 583  
SURFMORPHTOOL\_HPP 586  
SURFSUBDOMAIN\_HPP 589  
SURFVERTEXTREE\_HPP 591  
SURFVERTEXTYPE\_HPP 594  
SURROUNDTOOL\_HPP 553  
SWEEPTOOL\_HPP 596

## T

---

TD\_CELLINDEX\_HPP 598  
TD\_CENTER\_NODE\_HPP 600  
TD\_COPIED\_HPP 602  
TD\_DISTANCE\_HPP 604  
TD\_GENESIS\_ID\_HPP 606  
TD\_GEOMETRY\_SIZING\_HPP 608  
TD\_HEX\_KNOWING\_HPP 610  
TD\_LAYER\_HPP 614  
TD\_MORPH\_HPP 620  
TD\_NODEHARDLINE\_HPP 622  
TD\_NORMAL\_HPP 624  
TD\_PAVER\_HPP 626  
TD\_PILLOW\_HPP 628  
TD\_PROJECTION\_HPP 632  
TD\_SIZING\_HPP 634  
TD\_STRIDER\_HPP 636  
TD\_TIPTON\_HPP 645  
TD\_U\_V\_SPACE\_HPP 648  
TD\_VOL\_MAP\_HPP 649  
TD\_WEIGHT\_HPP 652  
TDLAPLACE\_HPP 612  
TDLPEDGE\_HPP 617  
TDSUBMAP\_HPP 642  
TIMER\_CLASS 653  
TOGGLIES 132

# Constants and Defined Macros Index

TOOL\_DATA\_HPP 661  
TOOL\_DATA\_USER\_HPP 664  
TOOL\_HPP 657  
TRANSLATETOOL\_HPP 666  
TREE\_HPP 669  
Treedeclare(name,typePtr) 669  
TRIANGLETOOL\_HPP 672  
TRIELEMENTTOOL\_HPP 673  
TRUE 132  
TWOHALFDIMTOOL\_HPP 680

## U

---

USE\_DYNAMIC\_MEMORY\_ALLOCATION 385  
USE\_VECTOR 645  
USERINTERFACE\_HPP 684

## V

---

VALIDITY\_CHECK\_HPP 685  
VOL\_VOID\_BOUNDARY\_HPP 702  
VOL\_VOID\_SEP\_TOOL\_HPP 704  
VOLMAPTOOL\_HPP 689  
VOLMESHTOOL\_HPP 691  
VOLSUBMAPTOOL\_HPP 697  
VOLUMEEDGEGETYPE\_HPP 699  
VOXEL\_INSIDE 193  
VOXEL\_INTERSECTS 193  
VOXEL\_OUTSIDE 193  
VOXELTOOL\_HPP 707

## W

---

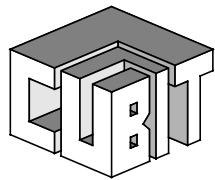
WARNING\_FLAG 159  
WHISKER\_CELL\_HPP 712  
WHISKER\_CHORD\_HPP 721  
WHISKER\_FACE\_HPP 726  
WHISKER\_HEX\_HPP 734  
WHISKER\_KNIFE\_HPP 738  
WHISKER\_SHEET\_HPP 752  
WHISKER\_WEAVER\_HPP 763

## X

---

XPATCHTOOL\_HPP 766





---

# Include File Index

## A

---

AcisGeometryEngine.hpp 589  
api/api.hxx 317  
ArrayBasedContainer.hpp 204, 246  
assert.h 184, 204, 213, 355, 361, 385, 435,  
514, 534, 721, 734, 752  
attrib/attrib.hxx 69, 75  
attrib\_gtc.hpp 71  
attrib\_snl.hpp 67, 73

## B

---

BaseHexer.hpp 338, 350  
box/box.hxx 62

## C

---

colors.h 88, 226, 243, 441, 507, 680, 752, 763  
CommandHandler.hpp 78, 284, 285, 295, 305,  
322, 389, 403, 451  
CubitBox.hpp 62, 477, 496  
CubitCollection.hpp 175, 547  
CubitContainer.hpp 127, 181  
CubitDefines.hpp 138, 193, 204, 264, 317,  
339, 367, 402, 406, 432, 454, 514, 516,  
518, 534, 536, 556, 559, 594, 610, 628,  
657, 661, 664, 702, 726, 738  
CubitEdge.hpp 338, 430, 589  
CubitEntity.hpp 97, 275, 298, 388, 477  
CubitFace.hpp 338, 589, 702  
CubitHex.hpp 291, 413  
CubitLogical.hpp 113, 430, 583  
CubitMatrix.hpp 586  
CubitMessage.hpp 168, 204, 514, 564  
CubitNode.hpp 213, 328, 338, 339, 430, 432,  
589, 610, 612, 620, 628, 642  
CubitNodeArray.hpp 378, 388  
CubitPlane.hpp 680

CubitString.hpp 62, 226, 243, 298, 317, 477,  
482, 510, 680, 684  
CubitVector.hpp 97, 126, 142, 145, 150, 154,  
166, 173, 193, 209, 226, 284, 322, 328,  
393, 441, 446, 464, 470, 477, 494, 499,  
507, 510, 524, 534, 553, 586, 624, 645,  
672, 680, 721, 734, 752  
CurveSubDomain.hpp 589

## D

---

DCubitEdgeArray.hpp 166  
DCubitFaceArray.hpp 113  
DCubitNodeArray.hpp 113, 622  
DEdgeUseArray.hpp 135  
DLBLEDgeLoopList.hpp 100  
DLBoundaryLayerEdgeList.hpp 100, 103  
DLBoundaryLayerFaceList.hpp 402  
DLBoundaryLayerList.hpp 103, 402  
DLChordPointList.hpp 524, 534, 712, 721,  
752, 763  
DLCopiedDataList.hpp 477  
DLCubitEdgeList.hpp 88, 145, 269, 393, 402,  
408, 446, 470, 494, 507, 589, 626, 697  
DLCubitEntityList.hpp 402  
DLCubitFaceList.hpp 88, 150, 180, 269, 302,  
367, 393, 402, 420, 441, 446, 494, 507,  
518, 553, 572, 589, 680, 689, 697, 702,  
704, 752, 763  
DLCubitHexList.hpp 142, 150, 339, 402, 408,  
507, 610  
DLCubitNodeList.hpp 62, 88, 97, 103, 150,  
269, 317, 328, 363, 378, 402, 413, 420,  
446, 470, 494, 507, 556, 573, 583, 589,  
622, 689, 697, 734  
DLCubitSheetList.hpp 402  
DLCubitStringList.hpp 480  
DLCubitVectorList.hpp 752

# Include File Index

DLcurveList.hpp 470, 494  
DLCurveSubDomainList.hpp 589  
DLDoubleList.hpp 94, 97, 376  
DLEDGEList.hpp 317, 470  
DLElementQualityList.hpp 454  
DLFACEList.hpp 494  
DLFaceUseList.hpp 142, 150  
DLFilePtrList.hpp 684  
DLGenesisEntityList.hpp 302, 402  
DLHexQualityList.hpp 343  
DLIntList.hpp 367  
DLList.hpp 281, 328, 393, 514, 669  
DLLoopList.hpp 113, 430, 589  
DLMeshEntityList.hpp 402, 752, 763  
DLNodeLoopList.hpp 113, 193, 378, 494, 572, 573, 707  
DLProtoHexList.hpp 88  
DLQuadQualityList.hpp 449  
DLRefBodyList.hpp 317, 402  
DLRefEdgeList.hpp 80, 100, 196, 269, 281, 284, 317, 376, 402, 417, 464, 494, 543, 586, 589, 673, 697  
DLRefEntityList.hpp 317, 402, 417, 496, 543, 583  
DLRefFaceList.hpp 103, 269, 317, 350, 376, 402, 417, 464, 507, 543, 596, 680  
DLRefGroupList.hpp 402  
DLRefVertexList.hpp 269, 281, 317, 402, 417, 464, 494  
DLRefVertLoopList.hpp 494  
DLRefVolumeList.hpp 302, 402, 417, 464, 494  
DLSelfCrossingList.hpp 518  
DLSheetChordList.hpp 752  
DLSideSetList.hpp 302  
DLStcEdgeList.hpp 559, 712  
DLSubLoopList.hpp 378, 572  
DLSubMapNodeList.hpp 567  
DLSurfSubDomainList.hpp 420  
DLSurfVertexList.hpp 80, 494  
DLTDSubMapList.hpp 378, 572, 642, 697  
DLVERTEXList.hpp 499  
DLVolEdgeList.hpp 507  
DLVolVoidBoundaryList.hpp 704  
DLWhiskerCellList.hpp 559, 712, 726, 734, 752, 763

DLWhiskerChordList.hpp 721, 734, 752, 763  
DLWhiskerFaceList.hpp 435  
DLWhiskerHexList.hpp 402, 712, 721, 752, 763  
DLWhiskerSheetList.hpp 402, 763  
DoubletPilloverTD.hpp 209  
DrawingTool.hpp 702  
DrawingToolDefines.h 135, 138, 142, 196, 226, 243, 255, 275, 350, 712  
DrawingToolInstance.hpp 408  
DynList.hpp 247

## E

---

EdgeMeshTool.hpp 308, 545, 562  
ElementBlock.hpp 258, 260, 261  
ElementQuality.hpp 342, 448  
ElementType.h 255, 281, 302, 470, 477

## F

---

face\_types.hpp 142  
FaceUse.hpp 441  
fstream.h 338, 684  
function\_templates.hpp 145

## G

---

GenesisEntity.hpp 255, 417, 543  
geom/transfrm.hxx 464  
GetLongOpt.hpp 684  
GridSearch.hpp 338

## H

---

hc.h 408  
HexTool.hpp 88, 209, 680, 763  
hoops\_driver.h 408

## I

---

idr.h 226  
iostream.h 320

## K

---

kernapi/api/api.hxx 317  
kerndata/attrib/attrib.hxx 69, 75

# Include File Index

kerndata/geom/transfrm.hxx 464  
kerndata/lists/lists.hxx 67, 73  
kernutil/box/box.hxx 62  
kernutil/vector/position.hxx 317

## L

---

limits.h 132  
LinearProgram.hpp 355  
lists/lists.hxx 67, 73  
Loop.hpp 589  
lpkit.h 361

## M

---

MapToolSupport.hpp 572, 578, 689, 697  
math.h 132  
MemoryManager.hpp 65, 120, 166, 246, 252, 275, 342, 376, 448, 459, 598, 604, 606, 608, 612, 620, 622, 624, 626, 634, 636, 669, 726, 734  
MeshEntity.hpp 94, 135, 142, 145, 150, 166, 180, 524, 721, 734, 752  
MeshIntersect.hpp 88  
MeshTool.hpp 250, 420, 583, 691

## P

---

PillowNode.hpp 339  
ProtoHex.hpp 338  
PyramidTool.hpp 443

## Q

---

QualityController.hpp 343, 449  
QualitySummary.hpp 343, 449  
Queue.hpp 539

## R

---

RefEdge.hpp 355, 589  
RefEntity.hpp 255, 269, 376, 402, 464, 470, 494, 496, 499, 507, 564  
RefEntityNameMap.hpp 480  
RefFace.hpp 589  
RefVertex.hpp 470, 589

## S

---

SDLControlPointList.hpp 608, 636  
SDList.hpp 480  
SelfCrossing.hpp 518  
SelfCrossingLoop.hpp 367, 372  
SheetEdge.hpp 90  
SLList.hpp 255  
spline/bs3\_crv/routines.hxx 103  
spline/bs3\_crv/sp3crtm.hxx 103  
spline/bs3\_curve/routines.hxx 103  
StarNode.hpp 213, 432, 628  
states.h 752  
states\_BaseHexer.h 88  
stdarg.h 159  
stdio.h 132, 288, 317, 766  
stdlib.h 132, 184, 382, 385  
string.h 117, 204, 226, 243, 246, 320, 402, 514  
SubDomain.hpp 196, 589  
SurfMeshTool.hpp 103, 113, 430, 572, 573, 578, 672, 673  
SurfVertexTree.hpp 80  
sys/types.h 123, 132, 653

## T

---

TDHexKnowing.hpp 213  
TDLPEdge.hpp 355  
TDNormal.hpp 626  
TDPillow.hpp 213, 610  
TDSizing.hpp 626  
Tool.hpp 226, 243, 288, 302, 317, 395, 409, 553, 707, 766  
ToolData.hpp 376, 567, 598, 600, 602, 604, 606, 608, 612, 614, 617, 620, 622, 624, 628, 632, 634, 636, 642, 645, 648, 649, 652, 726, 734  
ToolDataUser.hpp 388, 470, 494  
Tree.hpp 91, 537, 591  
TwoHalfDimTool.hpp 510, 596, 666

## V

---

vector/position.hxx 317  
VolMeshTool.hpp 346, 446, 689, 697, 704  
VoxelTool.hpp 573

# **Include File Index**

## **W**

---

WhiskerSheet.hpp 435

## **X**

---

x11/Intrinsic.h 226

x11/keysym.h 408

x11/xlib.h 226, 243, 408

x11/Xutil.h 408